Development of a Computer Aided Modelling System for Bio and Chemical Process and Product Design

Sales-Cruz, Alfonso Mauricio

Publication date:
2006

Document Version
Publisher's PDF, also known as Version of record

Development of a Computer Aided Modelling System for Bio and Chemical Process and Product Design

Ph.D. Thesis

A. Mauricio Sales-Cruz
CAPEC
Department of Chemical Engineering
Technical University of Denmark

March 29, 2006

CAPEC
Department of Chemical Engineering
Technical University of Denmark, 2006
ISBN 87-91435-37-4
Printed by Book Partner, Nørhaven Digital, Copenhagen, Denmark

Preface

This thesis is submitted as partial fulfilment of the requirements for the Ph.D. degree at Danmarks Tekniske Universitet (Technical University of Denmark). The work presented in the thesis was carried out in the Computer Aided Process-Product Engineering research Center (CAPEC) at Institut for Kemiteknik (Department of Chemical Engineering) from December 1st, 2002, to February, 2006 under the supervision of Professor Rafiqul Gani, and funded by Statens Teknisk-Videnskabelige Forskningsraad STVF (Danish Technical Research Agency).

The main goal of this thesis, entitled "*Development of a Computer Aided Modelling System for Bio and Chemical Process and Product Design*" has been the improvement and further development of an advanced computer aided modelling system, called ICAS-MoT, that aids the model developer in terms of model generation, model analysis, model translation, model solution and model validation/verification.

The main features of the developed system are highlighted with examples covering kinetic parameter identification of an anaerobic biogas process, the modelling and design of a short-path evaporator and the steady-state and dynamic simulation of a polymerization reactor, involving different aspects of mathematical model building, parameter estimation, and numerical and statistical analysis. In the next sections the major concepts and features of this package are reviewed.

Many of the areas studied in this thesis, such as solution of partial differential-algebraic equations or optimisation methods, really deserve much more attention. There is much more to be said about these topics than what is included in this thesis. However, the results presented in this thesis, in terms of the computer aided modelling tools, guidelines for systematic mathematical modelling building and parameter estimation problems, describe how to overcome many of the difficulties encountered in these problems.

This thesis consists of six chapters plus appendices: The modelling principles and modelling framework are given in chapter one. The current state of the art in computer-aided modelling is discussed in chapter two. Chapter three tackles the development of the computer-aided modelling system, while chapter four describes the software package that has been developed. Chapter five provides illustration of the applicability of the modelling software using several case studies and short examples. Finally, some general conclusions are drawn and a few guidelines for the future work are given in chapter six.

København,

Mauricio Sales-Cruz

Abstract

In the recent years, chemical process modelling has dominated the research and development from a process life-cycle perspective, with the need of providing a flexible set of computer-aided tools for model development and maintenance using an integrated environment. Such modelling tools should facilitate the (re)use of modelling knowledge during the process life-cycle, since the information incorporated in the models can be easily translated for different target applications such as steady-state simulation, dynamic simulation, process optimisation or experimental design.

Thus, it is useful to take advantage of the Computer-Aided Modelling Systems (CAMS), which must ensure the integration of existing tools and models into a software environment to support model definition, model evaluation, model analysis, model verification and model validation. An important modelling step is the model analysis that assures the model confidence, the input/parameter sensitivity, the model statistics, etc., using experimental data when it is possible. This step corresponds to a diagnosis based verification approach and may involve statistical analysis of the fitted (optimised) model parameters, which can help in the design of experiments with minimal effort.

The main contribution of this thesis is the development of a modelling framework and its corresponding software for systematic chemical process modelling, called ICAS-MoT, which has all the aforementioned features. It was specifically designed with focus on the structure and reuse of models and to facilitate model implementation. From the application perspective, the model description, model analysis, model identification (parameter estimation) and static dynamic simulation using the proposed modelling framework are shown and highlighted through several case studies.

A principal goal of this PhD-Thesis has been to use models for product and process design by testing and implementing them as fast as possible (in a reliable and efficient manner), writing the model equations without any programming effort, generating modules that can be used in another external software (via COM-object), and implementing several process/model configurations in the same environment. New features in the modelling language have been implemented to make it more powerful and easier to use, covering a wide range of applications.

Resumé på dansk

Udfra en proceslivscyklusbetragtning, har modellering domineret forskning og udviklingen indenfor den kemiske procesindustri i de seneste år. Dette har opbygget et behov for et fleksibel sæt af integrerede og computerbaserede værktøjer til model konstruktion og vedligeholdelse. Dette modelleringsværktøj skal muliggøre (gen)brugen af viden om modellering i processen livscyklus, hvor den i modellerne indbyggede information let kan blive omformuleret til beskrive forskellige ønskede anvendelser såsom stationær tilstand, dynamisk simulering, procesoptimering eller design af eksperimenter. Det kan derfor betale sig at bruge Computer-Aided Modelling Systems (CAMS), der skal sikre integrering af eksisterende værktøjer og modeller i en brugerflade, der støtter definering, evaluering, analyse og efterprøvning samt godkendelse af modeller. Et vigtigt trin i modelleringen er analysen, som skal give tiltro til modellen, samt sensitivitet i forhold til starttilstand og parametre, statistisk information etc., hvor der vil blive brugt rigtig (eksperimentiel) data, når dette er muligt. Dette trin svarer til en diagnosetest og kan involvere statistisk analyse af optimerede parametre, hvilket kan hjælpe i tilrettelæggelsen af forsøg ved en minimal indsats.

Denne afhandlings bidrag indeholder udviklingen af et modelleringsværktøj for systematisk modellering af kemiske processor kaldet ICAS-MOT, der indeholder de føromtalte egenskaber, da det specifikt er blevet designet med focus på struktur og genbrugelighed af modeller, samt muligheden for at implementere en model. Beskrivelse og analyse af modellen, parameter estimering samt statisk dynamisk simulering vil blive belyst fra et anvendelsesperspektiv i programmet ved hjælp af adskillige eksempler.

Hovedvægten er lagt på brugen af modeller i procesdesign ved at teste og implementere dem på den hurtigste måde (som er pålidelig og effektiv), opskrivning af modellens ligninger uden nogen form for programmering, generering af moduler til brug i andre programmer (via COM-objekter), og implementering af flere proves/model moduler i samme beregningsopgave. Nye elementer er blevet implementeret i ICAS-MOT, hvilket har gjort det mere kraftfuld og lettere at bruge, som dækker et stort anvendelsesområde.

Acknowledgements

I would like to acknowledge the persons that have directly contributed to this work, especially in the process of writing a thesis, since it is hard, hard work. The essential thing that made this possible was the support I got from those people around me. I would not have been able to write this thesis without the help and influence of many people throughout my life. On this occasion I would like to deeply thank the various people who, during the course of the work presented in this thesis, provided me with useful and helpful assistance and support.

First of all I would like to thank The Chemical Engineering Department at the Technical University of Denmark for admitting me into the PhD program and for giving me the opportunity to study abroad and learn about a different culture. To my supervisor, professor Rafiqul Gani, for the fruitful discussions, his inputs and suggestions during the development of this project. Special thanks to Martin Hostrup for his invaluable aid working out on coding the system.

Second, I would also like to thank all my friends and fellow PhD-students (Erik, Núria, Edgar, Vip, Piotr, Jan, Steen, Jakob, Florin and Bao-Lin — I hope not to forget anyone) for their moral support during the work on this project, but especially for the many hours that we have shared together during the past three years. All of you will be present in my memory and in my heart for ever.

Third, I want to express my gratitude to my parents for giving me a good education and teaching me that hard work is the key to success. Thanks also to my brothers and sisters who have been an inexhaustible source of love, inspiration and trust that even though the completion of a PhD project is a long process I were able to finish it.

Finally, and perhaps most importantly, I want to express my sincere gratitude to my wife, María Teresa, for her love and support during the different stages of this work and to my son, Alessandro (that is always in my heart). Thank you for enduring my absence while I was working on this thesis.

—A. Mauricio Sales-Cruz

Contents

List of Figures

List of Tables

1

Introduction to the Computer Aided Modelling

In general, chemical processes are characterised by a multi-stage set of specific chemical transformations with a significant number of components participating, that are connected by several process phenomena. As a result, complex mathematical models arise from mass, energy and momentum balances taking into account several considerations (kinetic, engineering, transport, etc.), which generate a large number of algebraic equation (AE), ordinary differential equation (ODE), differential-algebraic equation (DAE), partial differential equation (PDE), or partial differential-algebraic equation(PDAE) systems. These models play an important role in both research and practical engineering, as they are useful for process and product design, providing a better process understanding, parameter estimation, sensitivity analysis, process optimisation, dynamic simulation, planning and scheduling of process operations, diagnosis of process faults and so on. They provide the formal medium for describing the interrelationships among various variables and parameters, and their solution offers improved understanding of the static or dynamic behaviour of the system.

However, in the course of specific process engineering activities, the cost of developing the required models usually represents a significant part of the overall budget (Perkins and Barton 1987). It is estimated that process industries in the European Union spend approximately €300 M/year on modelling activities and as a result of this they gain more than €1B/year in added value (www.Sim-Serv.com). These benefits come in terms of optimising equipment size and plant throughput, reducing the expenses on scale-up and improving the quality of decision making. The above numbers are just an indication of the potential value that could be gained through modelling. However, advanced process modelling requires a lot of expertise, experience and software tools to be available. Therefore, much attention should be placed towards that direction. The central role of process modelling in all aspects of training and research in the areas of process/product design and operation is well recognised today. Although most of the earlier models used for training were for steady-state analysis, more recently emphasis has been on dynamic models as well. Recent years have witnessed the model-based approach being extended to the design of complex process/products, such as, membrane cells, drugs,

pesticides/insecticides, flavours/fragrances, and polymer systems.

Due to the large variety of chemical process units and physico-chemical phenomena as well as increasing requirements on the sophistication of models, the effort required for setting up a detailed mathematical model for a chemical process still remains high. Furthermore, a multifaceted family of models of varying degree of detail is often required in order to support the application of model-based techniques during the whole process life cycle (Bogusch, Lohmann and Marquardt 2001).

Since early efforts, computer-aided modelling has generally been organized around unilateral computations, which perform predetermined operations on fixed inputs to yield values for the desired outputs. This is still a common practice since it helps engineers and scientists to manage tedious calculations, coordinate and control diverse numerical tasks, by producing models that use computer resources very efficiently. The disadvantages of the traditional modelling approach (based on reflections in a series of inherent weaknesses) are listed below (Stephanopoulos, Henning and Leone 1990):

1. The time and costs associated with computer-model development are high.
2. The resulting models are difficult to document and maintain adequately.
3. The re-use of computer-aided models is minimal, they tend to be task-specific and are often intrinsically linked to solution procedures.
4. The models cannot be synthesized automatically by the computer in the course of automatic execution of an engineering task.
5. For interactive modelling the modeller is required to be highly skilled in programming.

As a result of these weaknesses the duplication of modelling efforts has been enormous. Accumulated modelling knowledge is almost impossible to use, since the underlying modelling context (purpose, assumptions, simplifications) has never been documented and rationalized.

So, why must every new modelling effort start from scratch (Vázquez-Román, King and Bañarez-Alcántara (1996))? Furthermore, automatic generation of models at higher abstraction levels cannot be done (for example, deriving models of an overall liquid-separation system from the models of individual separators). Finally, the fragmentation of modelling efforts and the specific goals of their computer implementation has often led to internal inconsistencies among the various models used in different process engineering tasks.

In order to overcome the modelling bottlenecks outlined above, a systematization of modelling as well as development of advanced computer-aided modelling environments are required. Moreover, with the current trend towards more highly computerised systems, it is useful to take advantage of the computer-aided modelling systems, developing solutions to such problems, reducing the

time consumption and investment costs, and achieving the product and process design successfully in a fast, reliable and efficient way.

1.1 Modelling Frameworks

Several research groups, mainly in Europe and North America have focused on the development of such kind of tools. Stephanopoulos and coworkers presented the MODEL.LA environment (Stephanopoulos et al. 1990)) which gave the first account of a modelling language specific for the chemical engineering domain. The recent reimplementation of the MODEL.LA environment (Bieszczad, Koulouris and Stephanopoulos 1999) provides a physicochemical phenomena-based modelling language for representing chemical process models and a modelling logic for constructing the underlying model. ASCEND is both a large-scale object-oriented mathematical modelling environment and a typed mathematical modelling language (Piela, Epperly, Westerberg and Westerberg 1991). OMOLA constitutes an object-oriented modelling language that allows one to define model libraries and to build models in a hierarchical manner (Mattson and Anderson 1993). DESIGN-KIT uses object-oriented programming to allow the development of models (Stephanopoulos, Johnston, Kriticos, Lakshmanan, Mavrovouniotis and Siletti 1987). MODDEV is a knowledge-based computer aided tool that assists the user in setting up the model equations that describe a chemical system. In MODDEV, process models are developed through aggregation of equation embedded building blocks (Jensen 1998). MODKIT supports the systematic development, maintenance, and reuse of chemical process models (Bogusch et al. 2001). DYLAN is an object-oriented environment for process modelling and simulation (Lund 1992).

Other researchers have proposed similar ideas (e.g. Asbjørnsen, Meyassami and Sørlie (1989); Sørlie (1990); Preisig (1995); Perkins, Sargent, Vázquez-Román and Cho (1996); Tränkle, Zeitz, Ginkel and Gilles (2000)) which lead to prototypical implementations of computer-aided modelling systems.

1.2 Need for Computer Aided Modelling Systems

This section provides the background for modelling of physical systems and illustrates the need for better tools and modelling languages. It is a motivating factor to know that an important part of engineering "know-how" is encoded in system models. This knowledge needs to be stored in a formal and reusable way.

1.2.1 The user requirements

Although there have been considerable advances in the field of modelling and simulation environments in the last decades (Pantelides and Brit (1995); Marquardt (1996); Bogusch et al. (2001)), currently existing modelling environments still do not sufficiently increase the productivity of modellers as pointed out by industrial modelling practitioners (Balci (1986), Geoffrion (1989) and Stephanopoulos et al. (1990)). An useful review given by Foss, Lohmann and Marquardt (1998) collects the requirements on modelling environments from a practitioners point of view. The key needs for computer-aided model building can be summarized as follows (Bogusch et al. 2001):

1. Models are not just equations. In addition to equations model representations should include model assumptions and limitations, information on the specification of the degrees of freedom and on model initialisation, etc. Furthermore, all decisions taken during a modelling project need to be recorded to render the modelling process transparent.

2. The development and storage of families of models for the same process need to be supported. As a consequence, the versions of a model which have been built during a modelling project (for whatever purpose) need to be documented together with their interrelationships.

3. Since many engineers have problems in formulating process models by writing model equations, the interaction between the modeller and the modelling tool must be moved from the equation to the knowledge . This not only allows user interaction based on chemical engineering concepts every engineer is familiar with, but also, forms a basis for the set-up of correct and reusable models.

4. Computer-Aided modelling tools should store and retrieve modelling knowledge to be used to guide the process of model development.

5. Support of model reuse and modification (for the same process after modification or for another similar process.)

6. A repository of predefined model building blocks of fine granularity like equations describing reaction kinetics or heat and mass transfer must be provided.

7. Automation of parts of the modelling process. Although model development largely is mainly a creative activity, automation of parts of the modelling process is possible. This includes knowledge propagation and documentation as well as report generation.

The above attributes are regarded as vital to a fully functional system developed to aid in the model building process. There are, however, further issues

that also need attention in the development of a computer-aided modelling system, for example,

1. Model calibration and validation.
2. The structured development and documentation of the models.
3. The ability to develop, analyse and solve large-scale models which integrate distributed, lumped and population balance system.
4. Model sharing and standardisation of the model, properties and solution interfaces in line with CAPE-OPEN initiatives.

These requirements are demanding and can only be fulfilled in the longer run. This is not only due to the significant implementation effort required, but also due to the need of further investigations of the fundamentals of computer-aided modelling.

1.3 Modelling Principles

This section provides the basic principles of the model building process, outlining the elements and procedures of a general systematic approach.

1.3.1 Model Definition

A model is an imitation of the reality and a mathematical model is a particular form of representation. In the process of model building the modeller translating the real world problem into an equivalent mathematical problem that is first solved allowed by attempts to interpret the results. This is done in order to gain insights into the original real world situation (Hangos and Cameron 2001).

Consider an object $\mathcal{M}$ that will be called a model of the object or phenomenon $\mathcal{P}$ if $\mathcal{M}$ is able to replace $\mathcal{P}$ so that investigation of $\mathcal{M}$ provides some information about $\mathcal{P}$ (Gershenfeld 1999).

Clearly, when building a model, it is necessary that certain characteristics of the actual system be represented by the model. These characteristics could include:

- The correct response direction of the outputs as the inputs change.
- Valid structure that correctly represents the connection between the inputs, outputs and internal variables.
- The correct short and/or long term behaviour of the model.

In some cases, certain characteristics which are unnecessary to the user of the model are also included. The resultant model has a specific region of applicability, depending on the experiments used to test the model behaviour against reality. These issues are more thoroughly investigated in the following section.

1.3.2 Model Attributes

The main failures that the model builder should be aware of, can be seen as the key attributes that might affect the modelling and analysis process. These are listed below (Arczewski and Pietrucha 1993):

1. Forgetting that modelling should serve some purpose and therefore that there must be a goal to modelling. Lack of awareness of this requirement makes it more difficult both for the modeller to do the work and for others to follow the reasoning in a scientific publication resulting from it.

2. Forgetting that the pragmatic aim of modelling is to produce the simplest adequate mathematical model. Quite frequently a simple mathematical model based upon rational foundations yields better results than a refined model that lacks solid foundations.

3. Lack of distinction between assumptions in the physical model and simplifications in the mathematical one. While the former should have an empirical basis, the latter is justified by the need to obtain simpler solutions. The permanent awareness of the type of simplification enables an easy improvement of the model.

4. Failure to check a model in terms of sensitivity of the response to changes in parameter values. If the model is highly sensitive, then it is of limited use for prediction purposes.

5. Forgetting that a model is only a simplification of the real world associated with the problem, and the model obtained cannot have any solution. This failure results in a deep belief that the solution to the model exists only because the model was developed for a phenomenon, whose existence may be known.

6. Conviction that obtaining a solution is just a question of accessibility to a computer having adequately large capacity and speed of operation. This failure may result in the generation of an immense amount of numbers, whose interpretation can be difficult or even wrong.

7. Tendency towards giving preference to the expected results. When a modeller has done everything possible to avoid prejudices in the model proposed by himself, he can always refer to a friendly criticism of colleagues.

8. Forgetting that a model is nothing but a model, and that its contribution may be subject to errors. So before exploring the computer calculations one should very carefully study the assumptions made and try to predict the qualitative features, such as stability of solutions to the model.

9. Unawareness that a model can be accepted as an adequate model only when it has passed the validation test. As long as the model has not been tested, the modelling process cannot be regarded as complete.

10. Lack of an inspection of the entire modelling problem when going into computational details. In order to produce an adequate model of a complex mechanical system one requires a wide understanding of many different topics. Sometimes one needs collaboration with people from different backgrounds.

In reality there are many doubts and even errors before the modeller arrives at a satisfactory model. It is important to keep some of the above characteristics in mind when developing a model for a specific application. In many cases it is clearly not a trivial issue. In other situations the model development can be quite straightforward.

1.4 The modelling process

To understand the process of mathematical modelling, consider the two worlds shown in Figure 1.1.

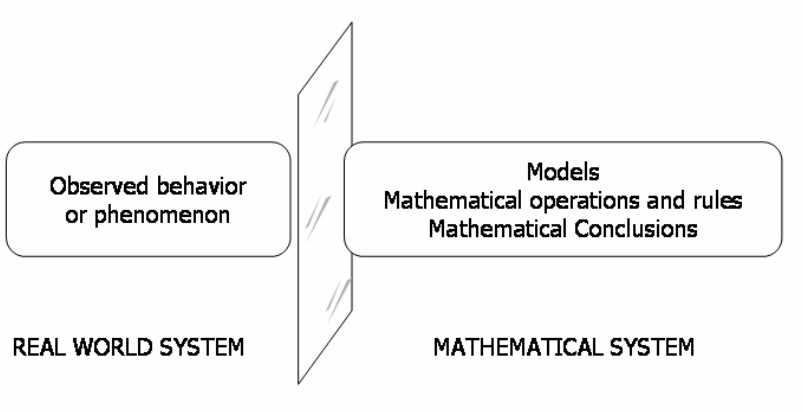


Figure 1.1: Characteristics of a real system and mathematical model

Consider a system where some behaviour or phenomena in the real world needs to be explained mathematically, in order to make predictions in the future and analyse the effects that various conditions have on it. One common way to obtain a mathematical explanation of the system under study is to perform

experiments and repeat them under different conditions in order to obtain an optimised experimental system that gives at the end, acceptable explanations about the real world (as shown in Figure 1.2).

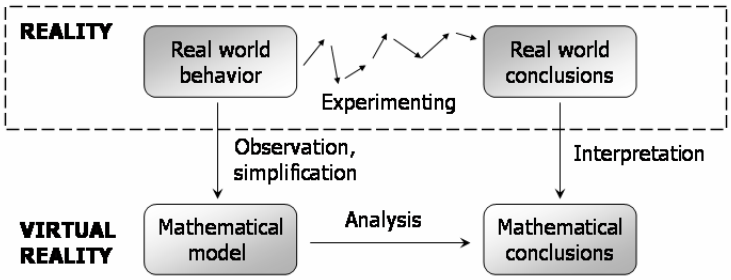


Figure 1.2: Physical phenomena (real world) and Mathematical abstraction (model).

An alternative way of reaching a better understanding of the real world is through mathematical modelling (as shown in Figure 1.2). According to this approach, first specific observations about the system under study is made, identifying the factors that appear to be important. As not all the factors involved in the behaviour can be considered, or even identified, some assumptions must be applied to simplify or eliminate some of these factors. Next, a rough model of the system behaviour is created based on the mathematical relationships among the selected factors. Having constructed a model, an appropriate model analysis needs to be made to get mathematical conclusions about the model. Note that these conclusions pertain only to the model, not to the actual real-world system under investigation. Next, the reproducibility of the mathematical model needs to be verified, because the model assumptions and the possible errors and limitations in the observations can generate anomalies in the inference of the system behaviour in the real world. In summary, the following rough modelling procedure can be considered:

1. Through observation, identify the primary factors involved in the real-world behaviour, possibly making simplifications.

2. Conjecture tentative relationships among the factors.

3. Apply mathematical analysis to the resultanting model.

4. Interpret mathematical conclusions in terms of the real-world problem.

Figure 1.3 describes the entire modelling process as a closed system.

Given, some real-world system, sufficient data is collected to formulate a model. Next, the model is analysed and mathematical conclusions about the

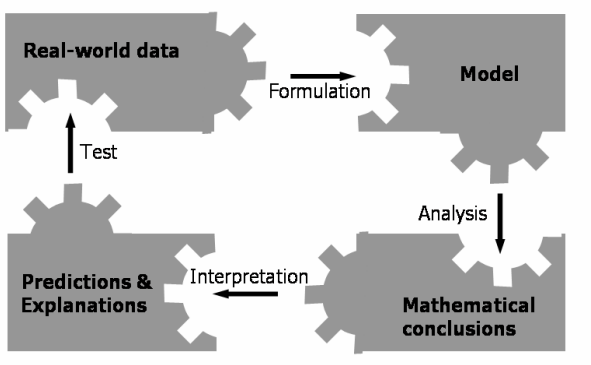


Figure 1.3: The modelling process as a closed system

system under study is reached. Then the model is interpreted and the predictions or explanations are offered. Finally, conclusions about the real-world system against new observations and data are tested. One may then find the need to go back and refine the model to improve its predictive or descriptive capabilities. Or perhaps one will discover that the model really does not fit the real world accurately, so a new model must be formulated.

The various components of this modelling process are studied in detail in the next sections.

1.4.1 The modelling approach

In the context considered here, the process of obtaining a mathematical abstraction of the system under study must finally result in a sort of systematic approach for process modelling, which is well-suited for computer implementation. This approach should comprise:

i) a systematic modelling procedure which supports both, the derivation of models from scratch, and the reuse and evolutionary modification of an existing model to meet the requirements of a new context.

ii) the decomposition of models and the definition of elementary modelling objects which can be aggregated to form a consistent model of (ideally) any chemical and biochemical process.

Before starting to set-up a process model, the problem definition should be clearly stated. This involves: definition of the process, the modelling goal, and the validation criteria.

1.4.2 The modelling Goal

All modelling is goal oriented. Models are built for a particular end-use and not to fill in idle time of the modeller (Németh, Cameron and Hangos 2005). This end-use influences the goal that the model must fulfil. It is important to establish the goals of the modelling activity, which at the outset, can be clear or potentially ill-defined. For example, the application areas of process design, control, optimisation or diagnosis usually lead to different model representations for the same physical system. Meeting the stated modelling goal provides a means of determining when the modelling cycle should be terminated (Lakner, Hangos and Cameron 2005).

1.4.3 Systematic modelling steps

Recently several authors (Hangos and Cameron (2001); Foss et al. (1998); Marquardt (1996)) have proposed systematic steps for the modelling process. Taking into account all of them, the generic modelling procedure can be summarised in ten steps, as described below:

1. System description (Model goal-set definition)
2. Problem definition (Model conceptualization and controlling factor identification)
3. Model construction or selection
4. Model analysis
5. Model data collection (need and source)
6. Model solution
7. Model verification
8. Model validation
9. Model implementation(model transfer)
10. Model documentation and maintenance

◇ **Step 1.** *System description.*
The identification of a problem is a phase needed because there is something that is not understood, or, a phenomenon that requires explanation. Typically this is a difficult task because of the difficulty in identifying a plausible mechanism and deciding what must be done.

◇ **Step 2.** *Problem definition*
This step refines the process description from step 1 by adding the modelling goal, and moreover, it fixes the degree of detail relevant of the modelling goal.

For instance, specifying: inputs and outputs, type of spatial distribution models (distributed (PDAE) or lumped (DAE)), the necessary range and accuracy of the model, and the time characteristic of the process model (steady-state or dynamic). According to Hangos and Cameron (2001), this step can be broken-down into the following two tasks:

(a) *Identify the controlling factors.* Here the physico-chemical phenomena that take place in the system must be identified. Typically reaction, diffusion (mass or heat), phase change (i.e. evaporation or condensation), heat conduction or radiation are the most common controlling factors.

(b) *Make assumptions.* Generally it is not possible to capture in a usable mathematical model all the factors influencing the problem that has been identified. The complexity of the problem can be simplified by reducing the number of factors under consideration, by neglecting some of the independent variables (for instance those variables whose effect may be relatively small compared to other factors involved in the behaviour), and by assuming relatively simple relationships.

◇ **Step 3.** *Model construction.*
Once a problem is identified and a mechanism proposed, one must formulate it mathematically. Often the difficulty lies in the choice of complexity: one would like to employ a simpler model, but on the other hand one should include every relevant process. The complexity of the model should depend on the final use of the model. Formulation involves equations and boundary conditions, and if the problem is a sensible representation of the physics, it will usually (though not always) be well-posed.

◇ **Step 4.** *Model analysis.*
In analysing a model its mathematical structure must be identified, one is often led through a sequence of similar types of calculation. To ensure that the model is well posed (i.e. the degrees of freedom are satisfied). To try to avoid certain numerical problems such as high index, having in mind that inappropriate specifications lead to problems with high index. In particular, it is often possible to break down a complicated model into simpler constituent processes (sub-models) which, for example, operate on different space and time scales.

◇ **Step 5.** *Model data.*
In the real modelling process, models are formulated using only first principles (white-box). However, mathematical models need experimental information to make them more predictive (grey-box models). Therefore, either measured process data or estimated parameter values are needed for the modelling process, but it is important to take into accound their uncertainties or precision. Moreover, at this point, it may be found out that there are neither suitable parameters values reported in the literature nor measured data to estimate them. This situation may force the modeller to reconsider the decision in Steps 1 and

2, and therefore go back to change them.

◇ **Step 6.** *Model solution.*
Having a solution is just the beginning of the analysis. It is possible to find obstacles in this step because the numerical computations cannot be established due to ill-posedness or stiffness of the equations. This difficulty can be tackled by some pre-treatment of the governing equations, for example, solving problems in dimensionless form. When the dimensionless process is done properly, the presence of small or large dimensionless parameters can be an indicator of singular perturbations, and thus stiffness. In many cases, this numerical inconvenience is an aid to analysis, facilitating the use of perturbation methods that can be used to gain insight into the solutions. In some other cases the model may consist of mathematical equations or inequalities that must be solved, or requires a best or optimal solution. Maybe one ends up with a model so unwieldy that it cannot be solved or interpreted. In such situations it is necessary to return to Step 2 and make additional simplifying assumptions, or sometimes to return to Step 1 to redefine the problem.

◇ **Step 7.** *Model verification.*
In this step the modeller needs to check whether the model is behaving correctly and if it was coded correctly (in the sense of syntax checking). The use of modular code and debug of the model equations can help to find mistakes in the coding process. Only some modellers rearrange the equation set before coding in order to improve robustness and efficiency during the numerical solution. Important issues are proper scaling, elimination of linear equations in the equation set or creating linear equations by introducing auxiliary variables for strongly non-linear problems, and reformulation of non-linear terms. In differential-algebraic equations problems some modellers try to eliminate algebraic equations to the extent possible for reasons of robustness . The index of a dynamic model is typically not considered explicitly, because higher index model formulations are usually (implicitly) avoided by an experienced modeller. This is particularly important for large-scale models.

◇ **Step 8.** *Model validation.*
Before the model is used, the behaviour against the reality must be checked. Before designing validation tests and collecting data, there are several questions that should be asked: First, does the model answer the problem identified in Step 1? Second, can one really gather the data necessary to operate the model? Does the predicted curve fit the experimental data? Third, does the model make common sense? Once the common sense test is passed, the model should be tested many times using experimental observations. Some tools to help to carry out this task include the use of sensitivity analysis to identify the key controlling inputs or system parameters, as well as the use of statistical validation tests. Usually, validation results indicate how to improve the model. For instance, one has to return to Step 1 and perform the step sequence again

if the developed model is not suitable for the modelling goal; or one has to return to Step 2 and reformulate the assumptions if the model predictions are reasonable over a restricted range of the independent variables but very poor outside those values. Note that it may not be possible to obtain the final model in one iteration through the modelling procedure.

◇ **Step 9.** *Model implementation.*
The model should be implemented such that the decision makers and users can understand, if it is ever to be used by anyone. Furthermore, unless the model is placed in a user-friendly form, it will quickly fall into disuse. Expensive computer programs sometimes suffer such a demise. Often the inclusion of an additional step to facilitate the collection and the input of data needed to solve the model determines its success or failure. Ideally, a mathematical model ends by returning to its origin.

◇ **Step 10.** *Model documentation and Maintenance*
Model documentation is one of the key requirements for effective CAMS environments and probably one of the least developed. The most important issues are: (a) the systematic recording of all the underlaying assumptions which are made during the course of the model development, (b) the documentation of the decisions or reasoning made to arrive at a particular model type, and (c) the generation of readable model descriptions and final reports for communication and archival purposes. Three types of documentation in current practice are: in line (i.e., in the model code), for end user or client, and on model development and application (including rationale and unsuccessful trials). Inadequacies in the area of documentation lead to major problems in model re-use or further development creating problems in maintainability. Regarding the model maintenance, remember that the model is derived from the specific problem identified in Step 1 and from the assumptions made in Step 2. So that, if the original problem has changed in any way, if some previously neglected factors have become important, or if any of the sub-models needs to be adjusted, then the model must be updated.

This generic modelling procedure highlights the iterative nature of model construction, but also emphasizes the importance of model simplification or model refinement as the result of the model validation procedure. If one cannot come up with a model or solve the one developed, one must simplify it. The model can be simplified by treating some variables as constants, by neglecting or aggregating some variables, by assuming simpler relationships or by restricting further the problem under investigation. On the other hand, if the model results are not precise enough, the model must be refined. Refinement of a model is generally archived in the opposite way: additional variables are introduced, more sophisticated relationships among the variables are assumed, or the scope of the problem is expanded. By trading-off between simplification and refinement, one determines the generality, realism and precision of a

model. This trade-off process cannot be overemphasized and constitutes the art of modelling.

The flow diagram shown in Figure 1.4 summarizes the methodology behind the modelling process and focusing on the aspects where the computer can help to the model developer.

Note that, like any model, this systematic procedure is an approximation process and therefore has its limitations.

The methodology shown in Figure 1.4 has certain advantages. It contains a blend of creativity with the scientific method used in modelling process. The first two steps are more artistic or original in nature. They involve abstracting the essential features of the problem under study, neglecting any factors judged to be unimportant, postulating relationships precise enough to help answer the questions posed by the problem and finally, yet simple enough to permit the completion of the remaining steps. Whereas these steps admittedly involve a degree of craftsmanship, one should apply scientific techniques (if available) to appraise the importance of a particular variable and the preciseness of an assumed relationship.

1.5 Objectives

The overall objective of this PhD project is the improvement and further development of an advanced Computer-Aided Modelling System (CAMS), called ICAS-MoT, and the demonstration through case studies how such a system can be used to solve interesting problems in product and process design, efficiently and reliably. More specifically, first the current state of the art in computer aided modelling will be reviewed. Then, based on the generic modelling procedure described in section 1.4, an existing computer-aided modelling system will be improved and new advanced modelling methods and tools will be developed to aid the model developer in terms of model (equation) generation, model analysis, model translation (to a computer language), model solution, model validation/verification, and finally, generation of model codes for direct use with external software. The new modelling features to be implemented will make the CAMS more powerful and easier to use, covering a wide range of applications such as: static and dynamic simulations of either lumped or distributed process models, parameter estimation (static optimisation), dynamic process optimisation, etc. Several case studies related to bio and chemical process and product design problems will be used to highlight the developed methods and tools. The case studies will require the modelling of complex processes and materials, an integration of models of different scales of size and time and validation against experimental data.

These objectives are discussed in greater detailed in Chapter 3, where the advanced computer-aided modelling system will be defined and justified.

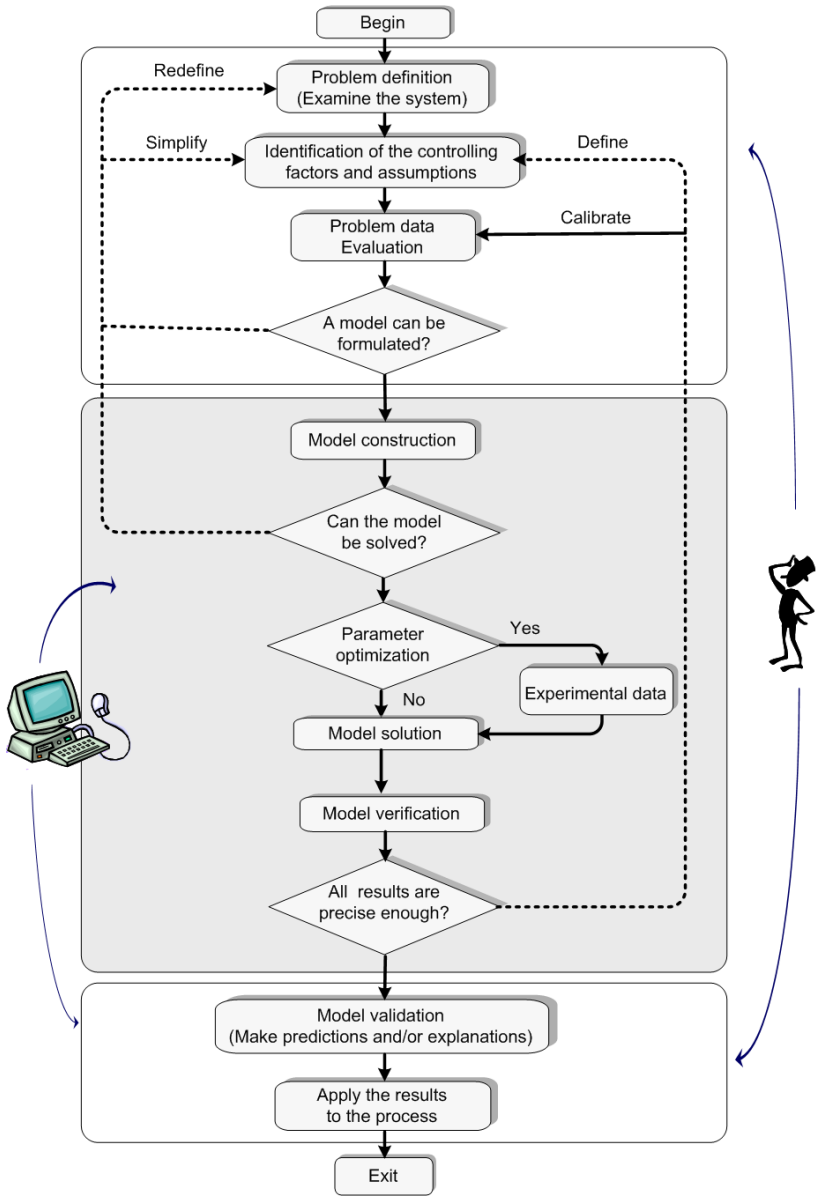


Figure 1.4: flow diagram for modelling process

1.6 Organization of the Thesis

The thesis comprises a total of six chapters organized as follows. The modelling frameworks and principles, the main modelling steps as well as a brief overview of the developed computer-aided modelling system are given in chapter one. Current state of the art in modelling, modelling frameworks, approaches and software used in the development of a computer-aided modelling system are presented in chapter two. This overview justifies and supports the main objectives of this PhD project. Chapter three discusses the development of a computer aided modelling system, establishing the main objectives required for a CAMS and the architecture behind of the developed modelling framework. As aforementioned, in this chapter the main modelling tools that a CAMS should have are discussed, defining which ones must be implemented to get a robust and reliable CAMS. Chapter four gives the formal descriptions of the modelling tools developed and incorporated to the system (ICAS-MoT), a software for CAMS. The way each modelling tool is used at each step of the generic modelling procedure is presented in detail in this chapter. The application of the main features of the developed CAMS are highlighted in chapter five through several short examples and case studies dealing with model parameter identification, steady state and dynamic simulation of open- and closed-loop processes, and process optimisation. Finally chapter six presents the conclusions, giving a summary of the main achievements and contribution of the thesis, followed by recommendations for future work.

2

Current State of the Art in Modelling

The state of the art as well as future trends in process modelling and simulation have been reviewed in the last decade in a number of contributions from different perspectives (e.g. Perkins and Barton (1987); Biegler (1989); Marquardt (1991); Boston, Brit and Tayyabkhan (1993); Pantelides and Barton (1993); Pantelides and Barton (1994); Jensen and Gani (1996); Hangos and Cameron (2001)).

In this chapter the status of the computer-aided process modelling systems (CAMS) and simulation is reviewed. The types of approaches available to aid in the formulation and solution of process models are described. Novel approaches and future trends are also presented. In terms of software development, the emergence of comprehensive and integrated process modelling packages in one super-platform is identified as a significant trend, with the potential to reduce the model development cost and increase the range of application of the current modelling frameworks. Finally the software used in the development of such modelling tools is described.

2.1 Modelling Methods/Approaches

Many currently available modelling systems have been reported in the literature, for example, Mattson, Elmquist and Otter (1998); Ponton, Gawthrop and Weiss (1999); Andersson (1994); Marquardt (1996); Perkins et al. (1996); (Preisig and Marquardt 2001); (Gilles, Gawthrop and Weiss 1998); Jensen and Gani (1996); Westerweele, Preisig and Weiss (1999); Tränkle et al. (2000); Bogusch et al. (2001).

Current CAMS tools can be classified into a number of categories although some may be classified to belong to several categories. Various general categorizations have been proposed by Bär and Zeitz (1990), Boston et al. (1993), Pantelides and Barton (1994), Marquardt (1996), Jensen (1998); Rico-Ramirez (1998) and Hangos and Cameron (2001). These modelling tools may roughly be classified into two main groups:

- block-oriented (modular or sequential) and
- equation-oriented (or simultaneous)

◇ *Block-oriented* (approaches mainly address modelling on the flowsheet level (stationary or dynamic)). Every process is abstracted by a block diagram consisting of standardized blocks which model the behaviour of a process unit or a part of it. All the blocks are linked by signal-like connections representing the flow of information, material and energy employing standardized interface and stream formats. Models of process units are preceded by a modelling expert and incorporated in a model library for later use. Modelling on the flowsheet level is either supported by a modelling language (i.e. ASPEN PLUS (Aspen Technology Inc.)) or by a graphical editor (i.e. Bär and Zeitz (1990)). In both cases, the end user selects the models from the library, provides the model parameters and connects them to the plant model. The incorporated chemical engineering knowledge as well as the model structure are largely fixed and not accessible. Common exceptions are physical property models which can be selected independently of the process unit model.

◇*Equation-oriented* modelling approach supports the implementation of unit models and their incorporation in a model library by means of declarative modelling languages [i.e. SPEEDUP (User Manual) (Aspen Tech.), ASPEN PLUS (Model Manager Reference Manual) (Aspen Tech.)] or by providing a set of subroutine templates to be complete directly in a procedural programming language [i.e. FORTRAN as in ASPEN PLUS (User Guide), (Aspen Tech.), DIVA (Kröner, Holl, Marquardt and Gilles 1990)] or gProms (Oh and Pantelides 1995)].

Each of the two approaches has advantages and disadvantages. *The modular approach* to modelling and simulation, though powerful and easily accessible to many engineers for the solution of standard flowsheet problems, does not adequately support the solution of more involved problems. This is largely due to the lack of models for many unit operations of adequate level of detail. Examples include multi-phase reactors, membrane processes, polymer reactors and most units involving particulates. Therefore, costly and time-consuming model development for a particular unit is often required during project work. On the other hand *Equation-oriented* modelling languages support the implementation of the models to a large extent. However competent utilization requires high effort and qualified personnel. Moreover they do not assist the user in developing models based on engineering concepts. Nor is there support for the documentation of the modelling process during the life-cycle of a process or for proper design and documentation of the model library. Reuse of validated unit models by a group of simulator users is therefore almost impossible and redundant modelling is unavoidable. The consistency and soundness of an initially even well-designed model library is inevitably getting lost over time (Marquardt 1996).

There are no different tools for the modelling expert or for the end user. Hence, modelling on the unit level requires profound knowledge in such diverse areas as chemical engineering, modelling and simulation, numerical mathematics, and computer science. The development of novel process unit models is therefore often restricted to a small group of experts.

2.1.1 Modelling Frameworks

The experiences described above has stimulated considerable effort in recent years in several research groups(Bogusch et al. (2001); Bezzo, Macchietto and Pantelides (2000); Lakner et al. (2005)). All the attempts (listed in Table 2.1) aim at facilitating model development and maintenance by enhancing the capabilities for model formulation, model reuse and adaptation as well as for maintenance and documentation. Ideally, the support should be extended to all phases of modelling, including the abstraction of the real process, the development of models from first principles and the symbolic manipulation of the model equations prior to numerical analysis. A large number of modelling tools exist for the construction, analysis and solution of mathematical models. Table 2.1 shows some of the most recent approaches and gives details of their main capabilities.

Common to all approaches is a multi-level modularisation of process models and a declarative (in the sense of explicit and symbolical) rather than a procedural (in the sense of implicit and algorithmic) representation. Each process, modelling tool provides its own model representation and model definition functions as well as its own solution algorithms, which are used for performing computer-aided studies for the process under consideration.

Overal, the developments can be classified into five groups (Marquardt (1996); Jensen (1998); Rico-Ramirez (1998); Hangos and Cameron (2001)):

- General modelling languages.
- Process modelling languages.
- Model expert systems
- Interactive (knowledge-based) modelling environments
- Flowsheet modelling environment

2.1.1.1 General Modelling languages.

These modelling languages can be seen as an extensions or further development of the class of equation-oriented simulation languages, where the user writes models in a structural language. They are designed to support hierarchical decomposition of complex models in order to facilitate reuse and modification

Table 2.1: Some recent advanced modelling tools

System Name	General modelling language	Process modelling language	Knowledge based system	Integrated with process simulator	Features AEs/DAEs	Features PDEs/PDAES	Features OPTIMISER	Reference
ABACUSS					*			Feehery and Barton (1996)
ACEND IV	*			*	*			Piela (1989). Piela et al. (1991, 1992)
Aspen customer Modeller		*		*	*			Aspen Technology Inc.
CHEOPS		*	*		*	*		Schopfer, et. al. (2004)
DYMOLA	*				*			Elmqvist (1978). Elmqvist et al. (1993) Cellier and Elmqvist (1993)
DYLAN	*			*	*			Lund (1992)
FEMLAB	*	*			*	*		The COMSOL Group
GAMS	*				*	*		Brooke et al. (1997)
gProms	*			*	*	*	*	Barton (1992). Barton and Pantelides (1994) Oh and Pantelides (1995)
HPT		*	*		*			Woods (1993)
KBMoSS		*	*		*			Vázquez-Román, et al. (1996)
ICAS-MoT	*	*	*		*	*	*	Sales et al. (2003)
MODASS		*	*		*			Sørlie (1990)
MODEL.LA		*	*		*			Stephanopoulos et al. (1990)
MODELICA	*							Modelica Design Group, http://www.modelica.org/
MODELLER		*			*			Preisisig(1991, 1992, 1994a, b)
MODEX		*	*		*			Åsbjørnsen et. al (1989), Meyssami and Åsbjørnsen (1989)
ModDev		*	*	*	*			Jensen (1996, 1998)
ModKiT		*	*		*	*		Bogusch et. al. (2001)
OMOLA	*			*	*			Nilsson (1993), Andersson)1994), Mattson and Andersson (1994)
PROFIT		*	*		*			Telnes (1992)
ProMoT	*	*	*		*	*		Tränkle,(2000)
SASE	*				*			Garrett and Hakim (1992)
SpeedUp		*		*	*		*	Sargent et. al. (1964), Perkins and Sargent (1982), Paloschi et al. (1983), Pantelides (1988)
SPL/SDL	*				*			Kiliccote (1996)
VEDA					*			Marquardt (1992a,b)
VERILOG		*	*		*			Thomas and Moorby (1996)

of existing models. They use concepts from semantic data modelling and object oriented programming. These languages are not restricted to chemical engineering applications, since the language definition is confined to a relatively small number of generic elements. Underlaying these general modelling language are the specific building blocks which are used to develop the user application. These are not necessarily in the form of modelling objects, but are languages objects suitable for writing equations. Examples of this kind of frameworks are ASCEND, gPROMS and MODELICA (see Table 2.1).

2.1.1.2 Process modelling languages.

The fundamental ideas of these languages are similar to the generic modelling languages. However, their language was designed to match the specific issues of a particular application domain. Typical examples are MODEL.LA and VEDA, where elements tailored to chemical engineering applications are included in the language definition. In such languages, very efficient modelling constructs exist, but their main limitation is that they have been developed to match the issues of specific chemical engineering applications.

2.1.1.3 Modelling expert systems.

The goal of these modelling environments is ideally to produce an adequate process model from a formal description of the modelling problem, initially provided by a user with a minimum (or rather no) interaction. As any expert system, it must consist of a knowledge base built on some hybrid knowledge representation formalism, a knowledge acquisition interface, an explanation facility, as well as, a separate reasoning (or inferencing) system which more or less automatically generates the model from a specification. MODEX has been a first attempt to create such a system. Also MODASS shows some signs of this general idea. After implementation and evaluation of a prototype both projects have been suspended. More recent developments drawing on expert systems ideas have been explored in PROFIT. Here, a detailed specification of the structural, as well as, the phenomenological characteristics completely defining an abstraction of the process under consideration is provided by the user. Based on these facts, a rule-based inference engine automatically determines a set of balance equations. MODDEV and MODELLER belong to this class too. In these cases, the modelling language is used to describe the model in process engineering terms, and as such is required to be flexible enough to encompass all possible process modelling situations. In most cases the user sees the model description in the language but does not necessarily write directly in that language.

2.1.1.4 Interactive (knowledge-based) modelling environments.

In contrast to autonomously acting expert systems, knowledge-based design environments or construction kits support the combination of elementary building

blocks to form an artefact. Since there are many distinct building blocks with a few restrictions confining possible combination, a very large number of valid configurations can be achieved. The core of such architecture is an interactive direct manipulative user interface which incorporates the modeller into the problem solving process. The system offers solution steps to be approved or rejected by a user rather than automatically solving the problem. Typically; the specification evolves together with the solution. Literally, there is no system as yet complying with this idea. Some characteristics, however, may be found in MODASS or in the knowledge based user interface of DIVA (Bär and Zeitz 1990).

2.1.1.5 Flowsheet modelling environments.

These environments are not specifically designed for the development of new models but rather for building integrated flowsheets from a library of predefined models covering a range of operation units and process units. Some facilities are available in certain systems to add new models such as the ASPEN custom modeller system for ASPEN (Aspen Tech.).

These systems are dominated by large commercial flowsheeting packages. These tools are the most widely used environments for carrying out large scale studies on process performance and dynamic behaviour. The level of the decomposition in this system stops at the unit block which can represent a process unit (i.e. reactor, mixer or distillation column). Table 2.2 gives a representative list of such systems currently available and gives an indication of their applicability to both steady-state and dynamic modelling applications.

Some of the modelling tools are integrated in computer-aided engineering environments to support modelling, analysis and eventually also synthesis of the process and/or its associated control system. To name just a few examples, OMOLA and the simulator OMSIM are integrated in a control system design environment (Andersson 1994). MODEL.LA is part of DESIGN-KIT, a computer aided process engineering environment (Stephanopoulos et al. 1987), an implementation of VEDA is currently integrated in the DIVA simulation environment (Marquardt 1996), the ASCEND modelling language is an integral part of an interactive modelling and simulation environment (Piela, McKelvey, and Westerberg 1992). Although all of these environments deal with a modular structure, neither of them is open in the sense of Baker, Chen, Grant, Jobling and Townsend (1993).

2.1.2 Some Novel Approaches

Recently novel modelling environments have been developed in order to support re-use of the existing models and to allow for combined use of different modelling frameworks in order to study high complex processes.

Table 2.2: Flowsheet Modelling Environments

Flowsheet environment	Feature		Reference	URL
	Steady State	Dynamic		
Aspen Dynamics		*	Aspen Technology Inc.	http://www.aspentech.com/
Aspen Plus	*		Aspen Technology Inc.	http://www.aspentech.com/
CADSIM Plus	*	*	Aurel Systems. Inc.	http://www.aurelsystems.com
CADSIM Plus		*	Aurel Systems Inc.	http://www.aurelsystems.com
ChemCad	*		Chempute Software	http://www.chemstations.net/
ChemPlant		*	ChemPlant Thecnology	http://www.chemplant.cz/
DESIGN II		*	WinSim Inc.	http://www.winsim.com/
DIVA	*		Holl, P. et. al (1988	
DYMON		*	Lund, P.C (1992)	
Hysys		*	AEA Hyprotech	http://www.aspentech.com/
ICASsim/DynSim	*	*	Jensen, A.K. and Gani R. (1996)	http://www.capec.kt.dtu.dk
PD-Plus			Prode	http://www.prode.com/
PetroPlan		*	Petroplan Control Systems Limited	http://www.petroplan.com/
Pro II	*		Chemstations, Inc.	http://www.simsci-esscor.com/
Prosim Plus		*	Bryan Research and Engineering, Inc.	http://www.prosim.fr
Simimica MB/CTL	*	*	Applied e-Simulators Ae-S	http://www.ae-sim.com/
SuperPro Designer	*		Intelligen Inc.	http://www.intelligen.com/
System7 Process Exp	*		EPCON International	http://www.epcon.com/

Two major issues have motivated the need for a framework integration solution within the context of chemical process modelling (Schopfer, Yang, von Wedel and Marquardt 2004). The first issue is that some process models consist of complex subsystems that cannot be modelled and simulated in a single general-purpose tool. The second issue is the need to support model re-use. When modelling a process at some point of its life-cycle, models for some process units are often already available in other modelling frameworks and consequently should be possible to use them.

Examples of these new frameworks are, CHEOPS (Schopfer et al. 2004), a tool-integration platform for chemical process modelling and simulation. Unit operation modules that contain the process models can be directly coded as a CHEOPS module or can also be taken from other model implementation frameworks such as gPROMS or ASPEN PLUS. ModKit (Bogusch et al. 2001) which allows model integration on a basis of conceptually consistent model building blocks. This framework is based on the ontological principles and was designed to capture all the available knowledge about mathematical modelling in chemical process engineering. ProMoT is an object-oriented and equation based modelling framework. It supports the development of modular structured simulation models. The construction of new models in ProMoT can be performed with a declarative modelling language or using a graphical editor that supports the construction of chemical plant flowsheets or cellular reaction networks.

In addition to the frameworks given above, there has been a number of other efforts at tools integration for modelling and simulation. The DIAS system is a flexible and extensive framework, which can integrate models provided by new or legacy software applications (DIAS http://www.dis.anl.gov/DIAS). DIAS has a very generic framework not bound to any specific domain. There are also efforts from scratch for specific domains. For example, FAMAS (FAMAS http://www.famas.tudelft.nl) is a distributed simulation architecture developed to support the operations and control of containers handling. TCAD (technology computer-aided design) is a framework that allows the use of heterogeneous process simulation tools for different process modelling steps in the manufacture of semiconductor devices to simulate the complete process. Other approaches for integrating modelling tools focus on realizing the integration of specific modelling tools and hence do not provide general solutions for the integrated model (Bezzo et al. 2000).

2.2 Software for CAMS Development

In novel modelling tools, implementation of the object-oriented formalism for model representation is accomplished by means of either an object oriented programming language (OOP) such as C++ (Horton 1998) or "Common LISP Object System" (CLOS) (Steele 1990), or by using a hybrid knowledge representation formalism like KEE (Fikes and Kehler 1985).

These languages provide (at least) some of the concepts used to construct the model representation formalisms and hence to facilitate implementation. However, FORTRAN has been one of the most popular computer languages used for science and engineering for almost 40 years. Even through much of the software developed in this period is now irrelevant and forgotten, a large portion still remains in use. In addition, new software is still being written in FORTRAN, simply because it is a proven and well established technology among scientists and engineers. Through the language's history of standardisation it has also become easily portable across operating systems, which is fundamental for software that often needs to run on many kinds of hardware, as well as survive several generations of computing trends.

Even though the FORTRAN language has developed and improved dramatically through the '66, '77 and '90 standards, it has failed to adopt the technique which today is recognised by many as the key to mastering complexity: Object Orientation. For these reasons, scientists and engineers are turning more and more towards languages that support OOP, and especially C++ which is by far the largest and most popular OOP language. Still, it would be too much of a revolution (and not a good idea) to throw away all the FORTRAN based software which has been tested and proven useful, especially if it can be shown that some of it (not all) actually fits quite well within the new OOP domain. Therefore a mixing of language between FORTRAN and C++ is widely used (Nyhoff and Sanford (1997)) .

One of the language that offers capabilities to mix language has rexxx increased attention and is becoming popular is Pyton (Pyton http://www.python.org) a portable, interpreted, object-oriented programming language. Python implementation is portable and it runs on several platforms.

Another important issue is the model re-usability. Usually, the modelling frameworks provide proprietary interfaces for the integration of models developed in other environments. In the EU project CAPE-OPEN and its successor Global CAPE-OPEN, standard interfaces for models and model-based applications were defined in order to achieve compatibility between different modelling tools (CAPE-OPEN http://www.colan.org).

Since process modelling tools may have different software implementations, a number of technical diversities have to be tackled to allow an integration platform to access different modelling tools. Consequently, the models provided by a tool may only be accessible via an application programming interface of certain language. This problem can be eased if a tool provides a component interface following certain middle-ware standards such as COM-object. Consequently, efforts may be required to enable the access to the models deployed in different platforms.

COM-Object is a software architecture to build component-based applications. COM objects are discrete components, each with a unique identity, which expose interfaces that allow applications and other components to access their features. They are completely language-independent and easily fit into an object-oriented programming.

2.3 Summary

As it was reviewed in this chapter, modelling approaches must allow various kinds of interpretation (either block-oriented or equation-oriented), such that steady-state and dynamic simulations (in terms of AE, DAE, PDE, PDAE systems) and also optimisation can be performed, and moreover support effectively model-based analysis and synthesis of processes. It has been seen that CAMS is an extremely important area in modelling practice. Conventional modelling frameworks provide some ability to develop or modify existing models. However, CAMS that specifically address the issues of model building, model analysis, model solution, model validation/verification, model integration, model reuse and model documentation and maintenance are vital for better uptake of modelling in the process life-cycle. Several approaches are developing these capabilities, some at the level of equation-based while others at really much higher conceptual level. As more and more new methodologies and numerical algorithms become available, modelling is expected to become the major bottleneck in the widespread use of model-based techniques in industrial and academic practice.

3

Development of a Computer-Aided Modelling System (CAMS)

The development of CAMS and simulation tools for chemical process and product design has advanced dramatically in the past twenty years. Furthermore, the perspectives for the near future developments are even more promising. The traditionally set path for the development of a general-purpose simulation package requires a CAMS expertise as an added value to the mathematical models of process units and physical properties evaluation. This modelling and problem solving expertise encompasses a wide variety of interdisciplinary fields such as numerical analysis for the solution of differential and algebraic equations, mathematical programming for the solution of the optimisation problems, and computer science in order to keep up-to-date to the capabilities that the computer technology offers today. This latter includes issues that range from hardware, to languages (FORTRAN, C++, pyton, etc.) and operating systems (Windows, linux, unix), to programming approaches (mathematical modelling vs. experts systems and artificial intelligence) and software structure (modular, equation-based approach, object-oriented programming).

Consequently, the development of a CAMS that can assist the model developer in a better and efficient manner continues to be the driving force for engineering practitioners and researchers investigating new methodologies or improving the already existing ones. As the size and complexity of process systems continue growing the use of CAMS is viewed as a solution. Effective use of CAMS however, requires a systematic approach that provides both a set of concepts for modelling and a set of guidelines for using such concepts.

This chapter states the objectives for developing a new CAMS, describes the architecture that will be employed and the methods and tools that are used by the CAMS.

3.1 CAMS Objectives

As pointed out in chapter one, process modelling is a fundamental activity in almost all process and product design and operations. Due to the large variety of chemical process units and physico-chemical phenomena, as well as, increasing requirements on the sophistication of models, the effort of setting up a detailed mathematical model for a chemical process is still very important. Furthermore, often a multifaceted family of models of varying degree of details is required in order to support the application of model-based techniques during the whole process life-cycle. In order to overcome this modelling bottleneck a systematization of the modelling together with the development of advanced computer-aided modelling environment is required. Therefore the main goal of a CAMS must be to assist the model developers in model building, model analysis, model solution and model documentation tasks in order to have in the modelling process life-cycle, more reliable and transparent mathematical models in a fast and efficient manner. Therefore, the CAMS must be gifted with some capabilities to handle different mathematical model formulations. These capabilities must include:

- Modelling based on AEs (steady-state simulation)
- Modelling based on DAEs (lumped and/or dynamic system simulations)
- Modelling based on PDEs (distributed systems simulation)
- Steady-state optimisation
- Dynamic parameter estimation

The CAMS must allow specifications of all five types of problems in a unified language (designed for process engineering applications).

To reach this goal, some existing computer-aided modelling tools will be extended and new advanced ones will be developed based on the ten-step generic modelling procedure described in chapter one. In fact, the first two steps (*problem specification* and *model conceptualization*) and the last one (*model documentation and maintenance*) are model developer-dependent and will be considered as "supplied by the user". The remaining seven steps can be aided by an appropriate CAMS and further developed in this project. Particularly, the computer-aided tools that will be improved or developed for each modelling step are:

- *Model data.* The data bank connection for compound properties constants and stream definitions will be added.
- *Model construction.* New tools as manipulation of matrices and conditional sentences will be included for the model equation generation.

- *Model analysis.* The existing tools (such as ordering of equations, setting of degrees of freedom, checking of singularity, etc.) will be improved and text messages will be provided to assistance the user during problem set up.
- *Model solution.* Some solvers will be extended (such as the backward integration), and two new solvers will be incorporated for solving: dynamic parameter optimisation problems and PDE systems (with automatic internal discretisation).
- *Model verification.* Tools to check that the code is correct (in the sense of syntax checking) will be improved: text messages providing assistance and the use of modular code and debug of the model equations helps in finding code mistakes will be added).
- *Model validation.* Two new tools will be incorporated to test and check the quality of a given model (including robustness and reliability): the statistical report (including the ANOVA, when it applies) and the sensitivity analysis.
- *Model transfer.* The tool for generation of COM-objects will be developed, so that ICAS-MoT models can be used in external software (such as Excel, Visual C++, Visual Basic and Fortran), but also can be transferred as customised ICASsim process unit library to be used in an ICASsim flowsheet.

3.2 CAMS Architecture

In order to have the proper architecture for the integrated computer-aided modelling framework an integrated environment, that allows for easy and effective data transfer and sharing of information among the different libraries, is needed. Figure 3.1 shows the modular CAMS structure.

The main components of the system are:

1. The Interface to the user (including the model editor)
2. The physical property database.
3. The thermodynamic library.
4. The solver library.
5. The simulation administrator.

These modules form the core of the CAMS. The user interface and the database access are usually implemented with the OOP language VC++, whereas the numerical part (thermodynamic and solver libraries) is usually programmed in FORTRAN.

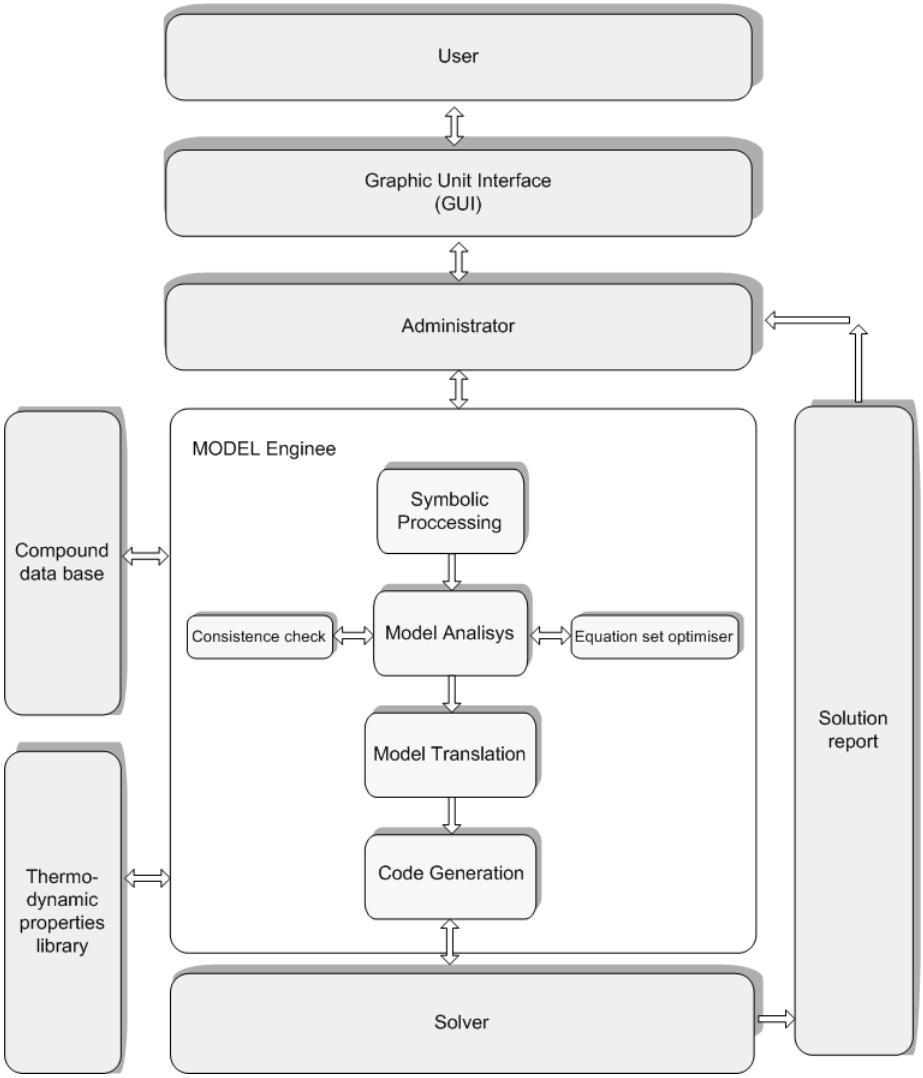


Figure 3.1: Sofware architecture of the modelling tool.

- *Graphic unit interface* (GUI). The unavoidable interaction between the modeller and the modelling tool must be facilitated by an interface, where the editor serves to define the problem. The flow of data is controlled from here. The connection of the different libraries are called from this interface and data are passed to and received from.

- *Compound data base library* (DB_Engine). A relational database could contain a variety of data, including: pure component properties, binary interaction coefficients, structural group description of molecules and experimental equilibrium data. The DB_Engine module needs to handle communication between the database and other libraries (see Tables A.7 and A.8 of Appendix A for a complete list of the properties). Parameters for the substances could be retrieved from in-house and commercially available physical property databases.

- *Thermodynamic library.* This is the module for thermodynamic calculations (i.e. physical properties, saturation point calculations, etc.). Table A.6 gives a list of the known thermodynamic functions. The CAMS framework should be able to combine thermodynamic model functions, and component properties within the same application model.

- *Solver library.* A solver library usually includes algebraic equations solvers, DAE-solvers and numerical optimisation methods. The appropriate solver for the model equations needs to be selected from the model engine library together with a corresponding solution strategy and then it must be sent to the solver library.

- *Model engine library.* This module is defined by the administrator of the CAMS, and it is in charge of control of the flow of information between all the procedures (i.e. thermodynamic library, compound data base, etc.), and of performing the main tasks related to the manipulation of the model equations. It is aided by an the equation parser that is used to tokenize the constituents of the model equations and to check whether the process quantities occurring in the equation are already defined. The process quantities found are linked with the equation. An attribute could indicate whether the equation is algebraic or differential or special function (see Table A.5). Moreover, arrays of equations can be identified. The array dimension can be specified symbolically, for example as the number of chemical components or reactions, and linked with the corresponding data in the data base library. Similarly equations may contain expressions that depend on symbolic expressions too (i.e. functions, special functions, external models, etc.). Figure 3.2 shows the parser structure.

  From the mathematical equation given as string characters, the parser performs a lexical and syntactic analysis and divides the string in tokens, after that an equation creator stores the tokenised equation and a translator prepares the equations to be solved (see section 3.3.3.1).

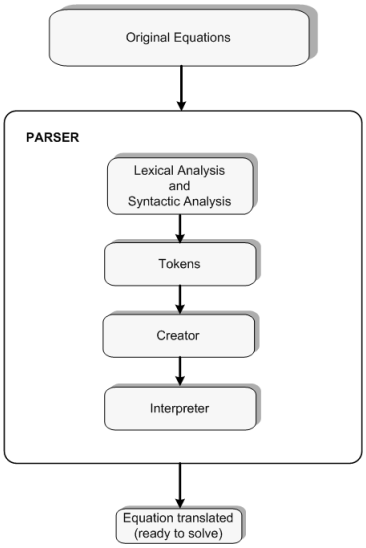


Figure 3.2: Equation parser structure

3.3 CAMS Methods and Tools

The computer-aided modelling system that has been developed belong to the group of *equation-based modelling* methods. However, the system developed crosses the boundary and surrounds the process modelling languages field, since it holds specific capabilities to handle mathematical models for chemical process. Moreover, the CAMS developed is able to transfer the equation-based process models to the simulation environment (e.g. ICASsim, Jensen and Gani (1996)). Therefore it can also be classified as flowsheet modelling environment.

As described in chapter two, a general modelling language is a system where the user can construct (write) a model in therms of mathematical equations using a declarative modelling language.

3.3.1 Method for Model Representation

A mathematical modelling language must contain knowledge about operands and operators and how they are put together in terms of variables and equations. This means that the possible building blocks that can be used to construct mathematical models are:

1. Operators
2. Variables (operands)
3. Equations (operators and variables)

4. Sub-models (set of variables and equations)

◇ *Operators.* The number of operands usually describes the kind of operators. Monadic or unary operators take one argument and dyadic or binary operators take two arguments. Table A.1 gives the list of the operators implemented in our CAMS. Mathematical functions are another kind of operators, one of the properties of these functions is that they return values commonly needed for operations on mathematical data. Some *special functions* have been implemented in this project in order to support equation-based models for chemical engineering applications. For a complete list of these functions see Tables A.4 and A.5 on appendix A.

◇ *Variables.* Variables are used to classify equations, analyse relationships and test the model validity. Several classes of variables are present in a mathematical model. These variable classes are described in Table 3.1.

Table 3.1: General variable classification

Variable type	**Meaning**
Parameter	Fixed variable (i.e. constants like: R, π, g)
Known	Variable given by the user (or connected system), it can change from one solution to another. (i.e. T, P, Feed stream)
Unknown	The value of this variable will be calculated by the algebraic solver (output variables).
Dependent	Integration variable, usually time-dependent.
Explicit	Variable on the RHS of an equation (i.e., the result of an equation evaluation).

RHS: right hand side

◇ *Equation.* Is the combination of variables and operators to produce a mathematical statement. This statement has two expressions separated by an equal sign. The expression on the left hand side (LHS) of the equal sign is explicit in terms of the expression in the right hand side (RHS). The LHS of an equation is generally a variable while the RHS could be an expression containing one or more variables.

Therefore the core of an equation-base modelling environment is the set of mathematical equations. As a mathematical equation consists of operands and operators (unitary and binary), a method to represent and store the equation must be provided. There is at least two different ways for equation representation:

1. *Expression tree.* An expression tree is a way to represent a mathematical expression. It consists of two different nodes: one called *leaf node*

(operands) that represents the variables and other one called *root node* that represents the operators. It takes care of all operators and their precedences. For example, consider the equation 3.1:

$$\underbrace{\frac{\partial u}{\partial t}}_{LHS} = \underbrace{\frac{\partial^2 u}{\partial r^2} + (3u+2)\left[\frac{1}{r}\frac{\partial u}{\partial r}\right] + \Phi}_{RHS} \tag{3.1}$$

The equation is entered as string character like 3.2:

$$\{partial(u,t) = partial(u,rr) + (3 * u + 2) * partial(u,r)/r + f\} \tag{3.2}$$

Where the word "*partial*" is used to represent the partial differential operator. The tokens identified are given in Table 3.2.

The equation creator converts this string into an expression tree by reading one term at a time and storing it into the tree. When a new term is read from the string, the creator first creates a new node and then adds the new node to the tree. A new node is created so that it has a pointer to its parent node and zero, one or two pointers to children nodes. Table 3.2 shows the corresponding expression tree decomposition for equation 3.2.

2. *Reverse Polish Notation* (RPN) is another way of expressing arithmetic expressions that avoids the use of brackets to define priorities for evaluation of operators. In ordinary notation, one might write

$$(1-2) * (3+4) \tag{3.3}$$

and the brackets tell us that we have to add 3 to 5, then subtract 2 from 7, and multiply the two results together. In RPN, the numbers and operators are listed one after another, and an operator always acts on the most recent numbers in the list. The numbers can be thought of as forming a stack, like a pile of plates. The most recent number goes on the top of the stack. In this notation the expression 3.3 would be expresed as:

1	2	-	3	4	+	*

An operator takes the appropriate number of arguments from the top of the stack and replaces them by the result of the operation.

Thus, the method used to represent model equations can be either the expression tree or RPN. Here a combination of both methods has been used. The

Table 3.2: Syntax Analysis: Tokens into (hierarchical) unit based on formal specifications for Eq. 3.2.

Token	**value**	**comments**
reserved word	*partial*	
identifier	u	dependent variable
reserved word	,	
identifier	t	independent variable
assignment statement	=	
reserved word	*partial*	
identifier	u	dependent variable
reserved word	,	
identifier	r	independent variable
form "("to ")"	*expression*	applied the syntactic recognizer
Multiplication sign	*	operator
reserved word	*partial*	
identifier	u	dependent variable
reserved word	,	
identifier	r	independent variable
division sign	/	
identifier	r	
plus sign	+	operator
identifier	f	

expression		
Expression	( )	
Digit	3	numerical value
Multiplication sign	*	operator
Identifier	u	dependent variable
plus sign	+	operator
Digit	2	numerical value

expression tree is used for model equation recognition and RPN method is used for internal storage since it has been found more convenient from a computational point of view (Whitney, Rode and Tung (1972).

3.3.2 Mathematical Models

Mathematical models for a process may be derived by applying the principle of conservation of mass, energy and/or momentum on a defined boundary (representing the process) and its connections to the surroundings. A process may be divided into a number of sections where each section is defined by a boundary and connections with other sections and the surroundings. In this way, models for different sections of a process may be aggregating together into a total model for the process. In general, the model equations may be divided into three main classes of equations:

- Balance Equations (mass, energy and/or momentum equations)
- Constitutive Equations (equations relating intensive variables such as temperature, pressure and/or composition to constitutive variables such as enthalpies, reaction rates, heat transfers, etc.)
- Connection and Conditional Equations (equations relating surroundings-system connections, equilibrium, controllers, etc.)

The appropriate model equations for each type of model may be derived based on the specific model needs. The model needs are translated to a set of model assumptions and together, help describe the boundary and its connections. Therefore, based on this description, a different version of a model for the same process may be derived. For example, a simple process model may include only the mass balance equations and the connection/conditional equations because the energy and momentum balance effects are assumed to be negligible and the constitutive variables are assumed to be invariant with respect to composition. A more rigorous model may include the mass and energy balance equations, the connection/conditional equations as well as the constitutive equations. There could be two modes of these models, steady state or dynamic. In the steady state mode, the rate of change of accumulation is assumed to be zero (or negligible) while in the case of dynamic mode, it varies with respect to time (the independent variable). An even more rigorous model may add the distribution of the intensive variables as a function of space (in one or more dimensions) (Sales-Cruz and Gani 2003).

3.3.2.1 The Model formulation

The general process formulation can be described in terms of :

Balance equations given by:

$$\frac{d\mathbf{x}}{dt} = f\left(t, \mathbf{x}, \mathbf{y}, \mathbf{p}, \mathbf{d}\right) \tag{3.4}$$

Constitutive equations:

$$0 = g_1\left(\mathbf{x}, \mathbf{y}\right) - \theta \tag{3.5}$$

and condition equation

$$0 = g_2\left(\mathbf{x}, \mathbf{y}, \mathbf{p}, \mathbf{d}, \delta\right) \tag{3.6}$$

In the above equations, $\mathbf{x}$ and $\mathbf{y}$ are usually regarded as the process variables in design while in control they are state and/or measured variables, respectively. These variables are usually the temperatures, pressures and compositions. $\mathbf{p}$ is the set of optimisation (design and/or manipulative) variables, $\mathbf{d}$ is the set of input (or disturbance) variables, θ is the set of constitutive variables (physical properties, reaction rates, mass/energy transfer rates), t is the independent variable (usually time) and δ is a time dependent controller variable. The steady state model is obtained by setting $d\mathbf{x}/dt = 0$. Otherwise, Eqs. 3.4 to 3.6 represent a dynamic model with a system of differential-algebraic equations (DAEs). Typically, there are $NC + 1$ balance equations with NC component balance equations and one energy balance equation per process (unit operation). The numbers of equations of the type of Eqs. 3.5 and 3.6 may vary considerably with the complexity of the model, the choice of the phenomena models, the controller type, etc. (Russel, Henriksen, Jørgensen and Gani 2002).

◇ *Algebraic Equation models.* The fundamental mathematical problem, the solution of which is demanded by steady-state simulation and design, is the solution sets of sets of non-linear algebraic equations of the form:

$$\mathbf{f}\left(\mathbf{x}\right) = \mathbf{0} \tag{3.7}$$

where $\mathbf{x}$ is a general vector of dimension n; $\mathbf{f}$ is a real valued set of functions of dimension $m \leq n$ arising from the physical model of a plant. In order for 3.7 to be solved, a set of $n - m$ specifications of the form:

$$\mathbb{M}\mathbf{x} = \mathbf{w} \tag{3.8}$$

Here $\mathbf{w}$ is a real vector of dimension $n-m$ and $\mathbb{M}$ is a $(n-m)$ x n specification matrix, such that:

$$\mathbb{M}_{ij} \in \{0, 1\}, \quad \sum_{j=1}^{n} \mathbb{M}_{ij} = 1 \quad i = 1, \ldots, n - m \tag{3.9}$$

Our equation-oriented CAMS can deal with different specifications of this kind of process models.

◇ *Differential Equation models.* The mathematical models of chemical engineering systems operating under transient conditions are usually described by mixed set of differential and algebraic equations of the form:

$$f(x, \dot{x}, y, u, t) = 0 \tag{3.10}$$

$$g(x, y, u, t) = 0 \tag{3.11}$$

here $x(t)$ and $y(t)$ are unknown vectors referred to as the "differential" and "algebraic" variables respectively, while $u(t)$ are "input" variables and are known functions of the time, t. Normally, the differential equation 3.10 arises from dynamic material, energy and momentum balances. Processes which are much faster e.g. thermodynamic equilibria or equations defining auxiliary quantities, yield algebraic equations of type 3.11.

Many DAE systems are very similar to the systems of ordinary differential equations (ODEs). In fact, if the algebraic equations 3.11 are solvable for the algebraic variables, y, given values of the differential variables, x, then the DAE system may be converted to the ODE (case of index = 1 system of DAES). The CAMS should be able to handle and solve both kinds of problem formulation: a pure ODE or DAE systems.

◇ *Dynamic parameter estimation* The mathematical models that arise from dynamic parameter estimation can be described by a system of DAEs:

$$D\dot{\mathbf{y}} = \mathbf{f}\left(t, \mathbf{y}, \theta\right) \tag{3.12}$$

and

$$\mathbf{y}\left(t_0, \theta\right) = \mathbf{y}_0\left(\theta\right) \tag{3.13}$$

in which $\theta \in \mathbb{R}^{n_P}$ is a vector of unknown parameters and $\mathbf{y} \in \mathbb{R}^{n_P}$ is a state vector depending on t and θ. $\mathbf{f}$ is, in general, a non-linear function that maps $\mathbb{R}\mathrm{x}\mathbb{R}^n\mathrm{x}\mathbb{R}^{n_P}$ into $\mathbb{R}^n$. D is assumed to be a constant n x n diagonal matrix with $d_{ii} = 1$ if the i^{th} equation is a differential equation and $d_{ii} = 0$ if the i^{th} equation is algebraic (Kristensen (2004)). The above notation for the DAE system is used for convenience to illustrate the basic set up of the parameter estimation problem. Later, when solution strategies are discussed, modifications to this notation are made depending on the specific method considered. In order to estimate the unknown parameters, a number of measurements are required for the process under consideration. Each measurement is characterized by a triplet:

$$(c_i, t_i, \tilde{y}_i) \qquad i = 1, \ldots, m \tag{3.14}$$

in which c_i indicates which component of the state vector $\mathbf{y}$ has been measured, t_i is the time of the measurement and $\tilde{y}_i$ is the measured value. m

denotes the total number of measurements. The solution of the model equations (3.12-3.13) for the c_i^{th} component at time t_i, which corresponds to the i^{th} measurement, is denoted by $y_{c_i}(t_i, \theta)$. With this notation the i^{th} residual is defined as:

$$r_i(\theta) = y_{c_i}(t_i, \theta) - \tilde{y}_i \tag{3.15}$$

This is a simplified statement, assuming that components of the state vector can be measured directly. More generally, the measurements are associated with the states through the measurement equation. When the true parameter vector θ^* is used, this equation becomes(Kristensen (2004)):

$$\tilde{y}_i = h(t_i, y_{c_i}(t_i, \theta^*), \theta^*) + \varepsilon_i \tag{3.16}$$

in which ε_i denotes the measurement error associated with the i^{th} measurement.

The method of estimation depends on the assumptions and knowledge about the measurement errors. Solutions methods that have been implemented in our CAMS are discussed in appendix B.

$\diamond$ *Partial differential equation models.* In chemical engineering, many process models are distributed, that is, defined by partial differential equations (PDE) in one, two or three spatial dimensions. This motivates the incorporation of PDE solvers (with automatic internal discretisation) into our CAMS to cover a wider range of chemical process models.

This system can be described in a general form by 3.17:

$$\frac{\partial \mathbf{u}}{\partial t} = \mathbf{f}(t, x, \mathbf{u}, \mathbf{u}_x, \mathbf{u}_{xx}) \qquad \begin{cases} \mathbf{b}(\mathbf{u}, \mathbf{u}_x) = \mathbf{z}(t) \ \textit{boundary conditions} \\ \mathbf{u}(\mathbf{x}, t_0) = \phi(\mathbf{x}) \ \textit{initial conditions} \end{cases} \tag{3.17}$$

It is obvious that the possible PDE problems described by Eq. 3.17 is sufficiently general to include many systems for which solutions may not exist or may not be unique. It must be, of course, the user responsibility to define a mathematically meaningful PDAE problem.

The PDE system (3.17) is completely defined if the following are given:

1. the number of PDE (N_{PDE})
2. the interval $[x_L, x_R]$
3. the initial time t_0
4. and the vector functions $\mathbf{f}$, $\mathbf{b}$, $\mathbf{z}$ and ϕ

One rather popular approach to the numerical solution of PDEs is the method of lines (MoL). It involves discretisation with respect to space variables first —thus generating a system of ODEs, which is typically large and stiff. The

linear systems arising in the course of stiff integration (with respect to the time variable) exhibit a typical block structure. For PDE systems in one space variable only, this linear system is usually solved by direct method such as a block tri-diagonal solver or band solver. For PDEs in more than one space variables, special iterative linear solvers appear to be exclusive choice. In frequent cases, algebraic conditions enter into the problem: examples of such an occurrence are 1-D formulations based on spherical or cylindrical coordinates or incompressible flow problems. In addition, part of the ODE may be implicit (DAE). Apart from any specifications of the applied linear solver, the described MoL approach directly leads to the PDE problems. Details of PDE classification and their solution procedure, as well as, a discussion about MoL are given in Appendix C.

Models represented by AE sets usually represent a steady state model; by DAE (ODE + AE) sets usually represent a dynamic model; while models represented by PDAE (PDE + ODE + AE) sets usually represent models in continuous domain. Solution of the model equations depends on the form of the model. Model forms of the AE-type require a linear or non-linear equations solver depending on whether the model is linear or non-linear with respect to the unknown variables. Usually, the AE set can be ordered and decomposed into subsets of implicit and explicit algebraic equations. The explicit equations can be solved analytically and this means that some models of AE-type may be explicit and solved analytically.

Models of the DAE-type may or may not include AE sub-sets. Most process models, however, include AE subsets, which when inserted into the ODEs, yield a system with ODEs only. Models of DAE-type may be solved in the dynamic-mode and/or steady state mode (where the condition under which the accumulation term becomes zero is sought). If the AE subsets are explicit, models of DAE-type they are usually solved in the ODE-dynamic mode while the DAE-dynamic mode is employed when a part of the AE subset is implicit. Models of the PDAE-type may or may not include AE subsets and ODE subsets. The PDE set is usually discretised with respect to the independent variables to yield a set of ODEs. Thus solution of PDAEs involves a discretisation step before solution of the resulting DAEs or AEs.

Note that introducing or removing model assumptions also generate different versions of a process model.

3.3.3 Model Translation and Analysis

The generation of process equations is just one step in a much larger set of computer-aided modelling tasks. It is often not the most difficult task to perform. There are key issues that must be addressed related to posing the problem in a final form that is solvable.

3.3.3.1 Model Translation

The model equations collected from the GUI have to pass through a syntactic and lexical recognition. The model translation process starts by reading the input line equation and parsing it, i.e. breaking it into its elementary components (tokens[1]) through which the line equation (source code) is transformed to the translated code (object) step-by-step.

The variables detected will be placed in variable classes (Table 3.3) each of which has one or more special impacts on the model. The translator applies the following rules: An equation item is a variable if:

- It is not an operator
- It is not a special operator
- It is not a mathematical function
- It is not a special function
- It does not start with a digit or a comma

The system works with different levels of variable classifications. The classification is used for both model analysis and model solution. When a model equation (equation 3.2, for example) is passed to the translator it is decomposed into two parts, the left hand side (LHS) and the right hand side (RHS). The split of the equation is always made from the location of the equal operator (=). Any equation must thus contain an equal sign in order to be used in the model. If an equation without a equal sign is entered, this equation will be ignored. The steps performed in the model translation are illustrated in Figure 3.3. It starts by scanning the text-based equations for validity to ensure mathematical consistency. After the equations have been validated, each equation and variable is expanded to match the current size of the problem, usually dependent of the number of compounds present in the system. The expansion step ensures that only the core model equations need to be specified and the model is not fixed or limited to a specific condition. Should the condition change, all the user needs to do is to repeat the translation step once more. Once the equations have been expanded, giving the actual equations to be solved, the variable classification step is invoked.

The classification step determines the base variables layer for each variable and set, per default, all RHS variables are classified as parameters. The user must afterwards reclassify the variables to suit the model needs. Table 3.4 gives the types used to classify the variables.

The node form of the classification system shows the top-level classification and the sub-level classification. A binary constraint will thus be a member of the LHS class, the explicit class, the constraint class and the binary class(Tables

[1]Linear scanning through character stream to form "tokens" is a character sequence with a collective meaning (identifiers, integers, reserved words, delimiters, etc.).

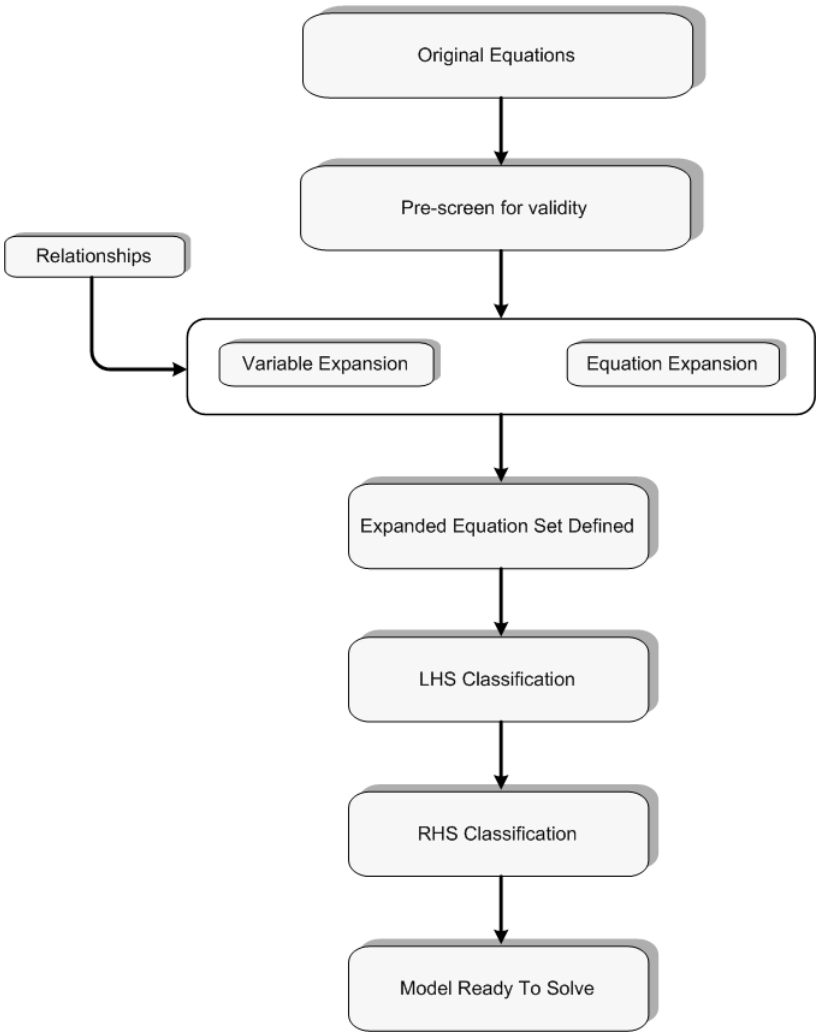


Figure 3.3: The steps performed in the model translation

Table 3.3: Variable classification levels class

Level 1	**level 2**	**Level 3**	**Level 4**
LHS	Explicit		
		Constraint	Real
			Integer
			Binary
RHS	Dependent Prime		
	Parameter		
		Manipulate (design)	
	Known		
	Unknown	Implict	
		Explicit	
	Dependent		

3.5 and 3.6). This classification scheme allows for a highly ordered system, which can be searched and manipulated very efficiently. The internal translation procedure will automatically perform the top-level classification. Each model equation explicitly determines if a variable is of the LHS class or the RHS class –It is thus not possible to manually select the top-level class since it is explicitly given in the model equations.

Subsequently the translated information (equations) must be stored using RPN, a class (*char Collection*) must be the root or base that will contain the complete equation and, each branch in the right hand side will correspond to the terms of the equation and the operators. Any combination of operators, parameters (constants), variables (unknown, known, dependent), mathematical functions, special functions and property models that is recognized as a valid expression is passed to the model translator engine for their analysis.

3.3.3.2 Model Analysis

The model analysis starts with the manipulation of the model equations so that they are transformed into a consistent set of equations that can be solved in a robust and efficient manner. Model analysis involves:

1. Degree of freedom (DOF) analysis
2. Determination of the structure of the equation system (incidence matrix)
3. Index Analysis
4. Partitioning and Ordering of the model equations
5. Determination of the sparse pattern
6. Analysing for numerical ill-condition expressions

Table 3.4: Variable type description

Variable Class	**Description**	**Notes**	**Variable Value**
Parameter	This variable has a fixed value, and will not be changed in the solution of the model.	Will be overruled if sub-classified as a manipulated variable.	Must be specified
Unknown	This variable is unknown and will be used to solve the residual equations in the model.	If the value is specified, it will be used as an initial value.	May be specified and will be used as initial guess.
Known	This variable is known and will not change during a single solution pass of the model.	The variable value may change outside of the model testbed, but will generally not be changed by MoT.	Must be specified
Explicit	This variable is the result of an equation. Can only change class to Dependent Prime.	The testbed will automatically detect . the variables of the Explicit class	Specified values will be overwritten during solution.
Dependent	Present only if the model is dynamic. If specified, the related variable in the Dependent Prime class must be given.	Y in dY/dt	Variable value represents the initial state of the model. Will change during integration.
Dependent Prime	Present only if the model is dynamic. Can only be chosen if one or more variables from the Dependent class are present. Variables of this class can only be selected from the Explicit class	DY/dt for Y	Function value for the Dependent variable. Specified values will be overwritten during solution.

Table 3.5: Level 3. optimisation Layer

Variable Class	Description	Notes	Variable Value
Constraint	Top-level description for constrain variables.	Must be sub-classed in level 4. Default value is Real	
Manipulated	This variable is the variable that will be manipulated by the SQP solver.	Can be a set	Bounds and initial value must be specified

Table 3.6: Level 4 Data Type

Variable Class	Description	Notes	Variable Value
Real	A constraint, which can take any values between the upper and lower bounds.	Can be a set.	Bound must be specified, value is calculated.
Integer	A constraint, which can take any integer values between the upper and lower bounds.	Can be a set.	Bound must be specified, value is calculated.
Binary	A constraint, which can either 0 or 1.	Can be a set.	Bound must be specified, value is calculated.

7. Analysing for stiffness of differential equations

Now a brief explanation of each point is provided.

◇ *Degree of freedom analysis*
The degree of freedom analysis (DOF) ensures that the model is properly posed and solvable. The basic concept of DOF is to determine the difference between the number of variables (unknowns) in the model, and the number of equations. Thus,

$$N_{DOF} = N_u - N_e \tag{3.18}$$

where N_{DOF} is the number of DOF, N_u the number of independent variables (unknowns) and N_e the number of independent equations. There are three possibles values for N_{DOF} to take:

1. $N_{DOF} = 0$. This implies that the number of independent variables and independent equations is the same. Therefore a *unique* solution may exist.

2. $N_{DOF} > 0$. This implies that the number of independent variables is greater than the number of independent equations. Therefore the problem is *underspecified* and a solution is possible only if some of the independent variables are "fixed" by some external consideration in order to reduce N_{DOF} to zero. In the case of optimisation these N_{DOF} variables will be adjusted to give a "best" solution to the problem.

3. $N_{DOF} < 0$. This implies that the number of independent variables is less than the number of equations. Therefore the problem is *over-specified* meaning that there are less variables than equations. The solution to such kind of problems is one that best "fits" all the equations.

When calculating the DOF for a process model, there are several schemes which can be used to account for the variables and associated equations(Hangos and Cameron (2001)).

Variables in the model. Typical variables in the model have already been discussed in section 3.3.3.1. Together with the variables there are parameters and constants which also appear. These parameters are normally fixed by the modeller but they can also be varied in order to optimise the process system. They could be included in the DOF analysis for completeness.

Equations in the Model. All equations must be independent. For example, one common mistake in writing mass balances is to write the individual mass component balances *as well as* the overall mass balance. These are dependent since the sum of the individual component balances gives the overall one.

Initial condition for dynamic models. It is important that the initial condition of *all* the differential variables is set by the user before any attempt is made to solve the problem. The initial conditions should not be included as a part of DOF analysis but should be assumed as essential to the correct establishment of the model.

Selecting variables to satisfy DOF. In large systems which are composed of a number of components or processing units, it is not valid to overspecify one unit and leave another underspecified so that the global DOF is satisfied. Hence, it is important to check that in large problems, the subsystems are not overspecified. As it has already been seen, dynamic model generally take the form:

$$\frac{dy}{dt} = f\left(y, z, t\right) \tag{3.19}$$

$$0 = g\left(y, z, t\right) \tag{3.20}$$

where y is the vector of differential or state variables $[y_1, y_2, ..., y_n]^T$ and z the vector of algebraic variables $[z_1, z_2, ..., z_q]^T$. f and g are general non-linear functions. The number of independent equation is then:

$$N_e = \dim\ f + \dim\ g \tag{3.21}$$

where $n = dim\ f$ is the number of independent differential equations and $m = dim\ g$ the number of independent algebraic equations. It is normally the

case that we have n states in the system with q auxiliary or algebraic variables ($q \geq n$). Therefore, the number of DOF in this case is:

$$N_{DOF} = n + q - \dim f - \dim g \tag{3.22}$$

Consequently, if $N_{DOF} > 0$, one has to specify: $n + q - \dim\ f - \dim\ g$ variables before any attempt can be made to solve the problem. The variables that are selected to be specified before solving the dynamic model are often called *design* variables or simple *specified* variables (Hangos and Cameron 2001).

◇ *Determination of the structure of the equation system (incidence Matrix)*
A useful tool for analysis of equation systems is the incidence or occurrence matrix. The incidence matrix tabulates equations vertically and variables horizontally, normally, an element in the incidence matrix is assigned a "1"' if the j^{th} equation contains the i^{th} variable, otherwise is assigned a "0". However, to provide additional information through the incidence matrix, our CAMS provides a visual and interactive means of analysing the equation-variable patterns in order to choose appropriate variables to satisfy the DOF or to investigate equation ordering to enhance solutions methods. Using a colouring code to identify if:

1. the variable is not in the equation
2. the variable is explicit with the equation
3. the variable is implicit with equation
4. the variable is known
5. the variable is a parameter (or design variable)
6. the variable is dependent
7. the variable is dependent prime

The incidence matrix gives a structural image of the model equations, and serves as a basis for determining the DOF, partitioning and ordering, index determination, etc. In cases when a system becomes complex, due to a large number of elements and a differentiated structure of connection between them (the system has an intricate topology) a graph reflecting the structure becomes an extremely valuable modelling tool.

◇ *Index analysis*
The index of a DAE is defined as the minimum number of times that part of a DAE (the AE) must be differentiated with respect to the independent variable in order that the algebraic system of equations has to undergo to convert the system into a set of ODEs (Brenan, Campbell and Petzold 1989).

The index of a pure ODE system is zero by definition. If the index of a DAE is 1, then the initial value of the differential variables can be selected arbitrarily, and the DAE set can be easily solved by conventional methods such as Runge-Kutta (RK) or Backward Differentiation Formula (BDF) methods (Hangos and Cameron (2001)). If however, the index is higher than 1, special care should be taken in assigning the initial values to the variables, since some "hidden" constraints lie behind the problem specification.

The following procedure can be used to determine the index of an equation system (Jensen 1998):

1. Remove all parameters, known variables, and the independent differential variables from the incidence matrix.

2. Assign the explicit variables to their respective explicit equations and remove them from the incidence matrix.

3. Assign the dependent differential variables to their ODEs and remove them from the incidence matrix.

4. Screen the remainders of the incidence matrix for equations containing only one single "unknown" variable. If found, remove these equations an their associated "unknown" variables from the incidence matrix. Repeat until no more equations with only one single "unknown" variable exist. Note that the state of the variables that are removed during this procedure must be "unknown" in order to have an index-zero problem. If it is nor acceptable to assign a variable to "unknown", leave the variable and its associated equation within the incidence matrix.

5. The states of the remaining variables in the incidence matrix have to be set so that the matrix between the remaining equations and the remaining "unknown" variables is regular. If this is not possible, the system is either an index-one or higher index problem.

6. If the index is different from zero, the actual index can be determined by applying the following twofold procedure (Brenan et al. 1989):

   - Reformulate the DAE so that it can contain algebraic constraints that are explicit in the dependent variables.
   - Differentiate the algebraic constraints with respect to the independent algebraic variables.

   Every iteration in this twofold procedure reduces the index by one. Thus, this process is continued until an explicit ODE system is obtained, and the number of iterations used to obtain an explicit ODE system is equal to the index of the DAE model.

In general, index problems arise when extensive state variables are either explicitly or implicitly constrained by algebraic relationships. However, numerical routines are generally incapable of handling these high index situations.

◇ *Partitioning and Ordering of the model equations*
The partitioning and ordering of an equation system are the factors that distinguish the different simulation strategies, i.e., the simulation strategy is determined by way the model equations are grouped, and order in which the groups are solved. Regarding the order, in which the model equations are solved, the following issues are central:

1. Sequential solution procedure. The model equations are solved in a specific order.

2. Simultaneous solution procedure. The implicit model equations are solved together.

Because iterative solution procedures depend on the partitioning and ordering of the model equations, some simulation strategies solve a particular set of equations better than others. Thus, the question is which simulation strategy is the most appropriate?

When the incidence matrix is available, the applied simulation strategy should be:

1. Evaluate all the explicit algebraic equations.

2. Determine and solve the linear subset of the remaining implicit algebraic equations.

3. Determine the block structure of the incidence matrix of the remaining implicit non-linear algebraic equations.

4. Converge each of the independent equation blocks individually with either the simultaneous approach or the sequential approach.

After the explicit algebraic equations have been evaluated, the subset of residual equations that are linear in unknown variables should be identified and solved. The reason for solving the linear equations separately from the remaining implicit algebraic equations is that the linear algebraic equations can be solved with one iteration, which means that the number of equations will need to be converged through iteration is reduced.
After the explicit and the linear algebraic equations have been removed from the equation system, the groups of equations that can be converged independently of other groups of equations should be identified. That is, the block structure of the equation system should be determined. For this purpose, the incidence matrix can be applied.

Finally, for each independent equation block, a decision regarding converging the block simultaneously or sequentially must be made. A requirement for applying the sequential approach is that the DOF must be satisfied within each of the sequentially solved blocks. This makes the simulation approach more flexible with respect to change in variable specification.

◇ *Determination of the sparse pattern*
A sparse (thinly populated) matrix is a matrix where most of the elements are zeros. The goal of introducing special storage and special methods for sparse matrices is to avoid superfluous storage and calculations. Sparse matrices can either have a structure pattern, like band or block-diagonal matrices, or they can have random structure. Matrices with random non-zeros are stored using a coordinate representation, a row-ordered list representation or a column-ordered list representation. For sparse matrices with a structure pattern, special efficient storage methods can be applied. When the equation-oriented solution approach is applied, the application of sparse methods becomes of special importance. Usually, if the number of non-zero elements in the Jacobian matrix is less than 25% of the total number of elements, CPU-time can be saved by applying sparse methods. However, to apply sparse methods, a sparse pattern is needed. Simulation engines that are able to use sparse methods are often able to estimate the sparse pattern numerically. This is done by performing a numerical determination of the Jacobian and applying the assumption that the elements that are less than a specified value are equal zero. This estimation method, however, is expensive with respect to CPU-time and there is a risk that the estimated pattern is not correct. Therefore, it is recommended to provide the analytically determined pattern, which is directly given by the incidence matrix.

◇ *Analysing for numerically ill-condition expression*
A mathematical expression that cannot be evaluated numerically is called ill-conditioned. It is therefore desirable that the process-modelling tool is able to detect their existence and replace them with numerically well-behaved expressions. Sometimes these ill-conditioned expressions can be replaced with a variable. For example if an expression contains $1/x$, then the substitution $y = 1/x$ can be performed and the system can be solved for y instead of x. Removing ill-conditioned expressions by variable substitution is possible if one of the variables within the ill-conditioned expression only occurs within the ill-conditioned expression and not in other parts of the equation system.

◇ *Analysing for stiffness of differential equations*
One of the major problems that arise in process modelling is the issue of stiffness. This characteristic of the model puts significant demands on the numerical techniques that can be used to solve such problems. The characteristics of stiffness can be seen in a number of ways:

1. the "time constants" of the process
2. the eigenvalues of the model equations.

Many physical systems display a type of behaviour characterized by a wide range of time constants in the system. The time constants reflect in some way how slowly or quickly a particular component or phenomenon reacts to a disturbance in the system or to the action attributed to it. For example, in a reaction system where we have a CSTR (Continuous Stirred Tank Reactor), there may be reactions which occur very quickly, whilst others are very slow. The fast reactions are associated with small time constants whilst the slow reactions are related to large time constants of the system. These time constants are related to the eigenvalues of the system equations as:

$$\tau_i \propto \frac{1}{\lambda_i} \tag{3.23}$$

where τ_i is the time constant and λ_i is the real part of the eigenvalue of the Jacobian matrix given by $\partial \mathbf{f}/\partial \mathbf{y}$. Note that small time constants correspond to large eigenvalues and vice versa. Where there is a system with a mixture of widely varying time constants, ill-conditioned behaviour is expected. It is common to quantify the degree of stiffness by the stiffness ratio, which is defined as (Cameron and Gani (1988)):

$$S(t) = \frac{\max\limits_i |\mathrm{RE}(\lambda_i)|}{\min\limits_i |\mathrm{RE}(\lambda_i)|} \quad \mathrm{RE}(\lambda_i) < 0, \qquad i = 0 \ldots N_e \tag{3.24}$$

However, this definition is poor when the real part of some of the eigenvalues are close to or equal to zero. When a stiff problem is presented, an implicit integrator should be used, because their stability area is larger than the explicit methods. Thus, the stiffness ratio can be used to select the numerical method of solution.

3.3.4 Model Solution

The general strategy in the the solution step is given in Figure 3.4.

Once the desired solution strategy has been selected, the model equations can be solved.

Depending on the selected strategy, the administrator, which drives the solution, will instantiate one or more solvers from the library. Before the solution starts, the CAMS needs to provide the administrator with an ordering filter, which ensures that the equations are solved in the correct order.

This procedure is divided intro three parts:

1. *Administration.* This part drives the solution procedure.
2. *Solution.* Using the text-based equations, this part solves the translated equations.

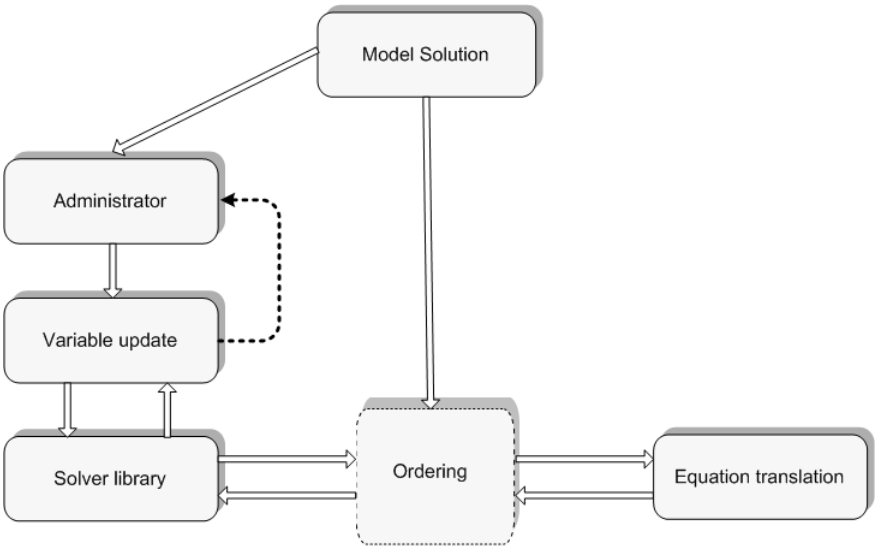


Figure 3.4: Model solution structure

3. *Solver perturbation and iteration.* This part is linked to the solver library and could, for instance, use a Newton-based technique to determine the unknown variables of a process model.

The administrator of the solution procedure can be divided into two parts:

- A global administrator
- One or more local administrators

The global administrator handles all solution modes by setting up the overall problem, verifying that the required information is present and performing small miscellaneous tasks. It also allocates the memory required for the task and spawns itself in a thread to allow for the GUI updates and possible multiprocessor solution. The global administrator creates one local administrator, its class determined by the top-level problem definition (algebraic problem, dynamic problem, optimisation problem). The local administrator handles a particular part of the solution itself. Each local administrator can solve a given problem by itself and can spawn other local administrators if required by the model solution.

Each local administrator instantiates one or more solvers from the solver library. The possible local administrators are:

- Algebraic administrator
- Dynamic administrator

Table 3.7: Solver structure

Administrator	**Action**
Global administrator	Created by ICAS-MoT to obtain the solution
Local dynamic administrator	Overall solution strategy is solving a dynamic model. Created by Global administrator
Local algebraic administrator	Solution mode is ODE, IAEs are present, i.e. an algebraic solver is required. Created by Local dynamic administrator

- Optimiser administrator

The global administrator combines all the required local administrators needed to solve the given problem and handles the overall solution sequence. The local administrators are only connected to the model via the global administrator.

The ordering provides the information on which partitions are present in the model. This information is used by the local administrators to increase the efficiency of the model solution, by only solving the relevant part of the problem at given point in the solution.

Figure 3.5 shows the full solution algorithm for an algebraic model.

A differential model (without optimisation) can be categorized in 3 cases (index 1 or lower):

- No Implicit Algebraic Equation systems (IAEs) present
- IAEs present, solve sequentially. ODE mode
- IAEs present, solve simultaneously. DAE mode.

The first case has only Explicit Algebraic Equations (EAEs) and no IAEs and requires only a local dynamic administrator in order to solve the problem. The second case, ODE mode, has IAEs and requires a local algebraic administrator (solving IAEs) as well as a local dynamic administrator (solving ODEs). The third case, DAE mode, has IAEs present, but these are solved simultaneously with the ODEs and thus requires only a local dynamic administrator. Due to the simultaneous approach in the DAE mode, the IAEs must be properly initialised in order to obtain a robust solution. Usually, the DAE mode is solved in two parts. The first part is solved in ODE mode to initialise the IAEs and the solution will then be switched to the second part which is solved in DAE mode. This will ensure perfect initialisation of the IAEs and a robust solution.

The algorithm for solving a DAE problem in ODE mode is shown in Figure 3.6. Note that the box in bold is actually the local algebraic administrator, shown in Figure 3.5.

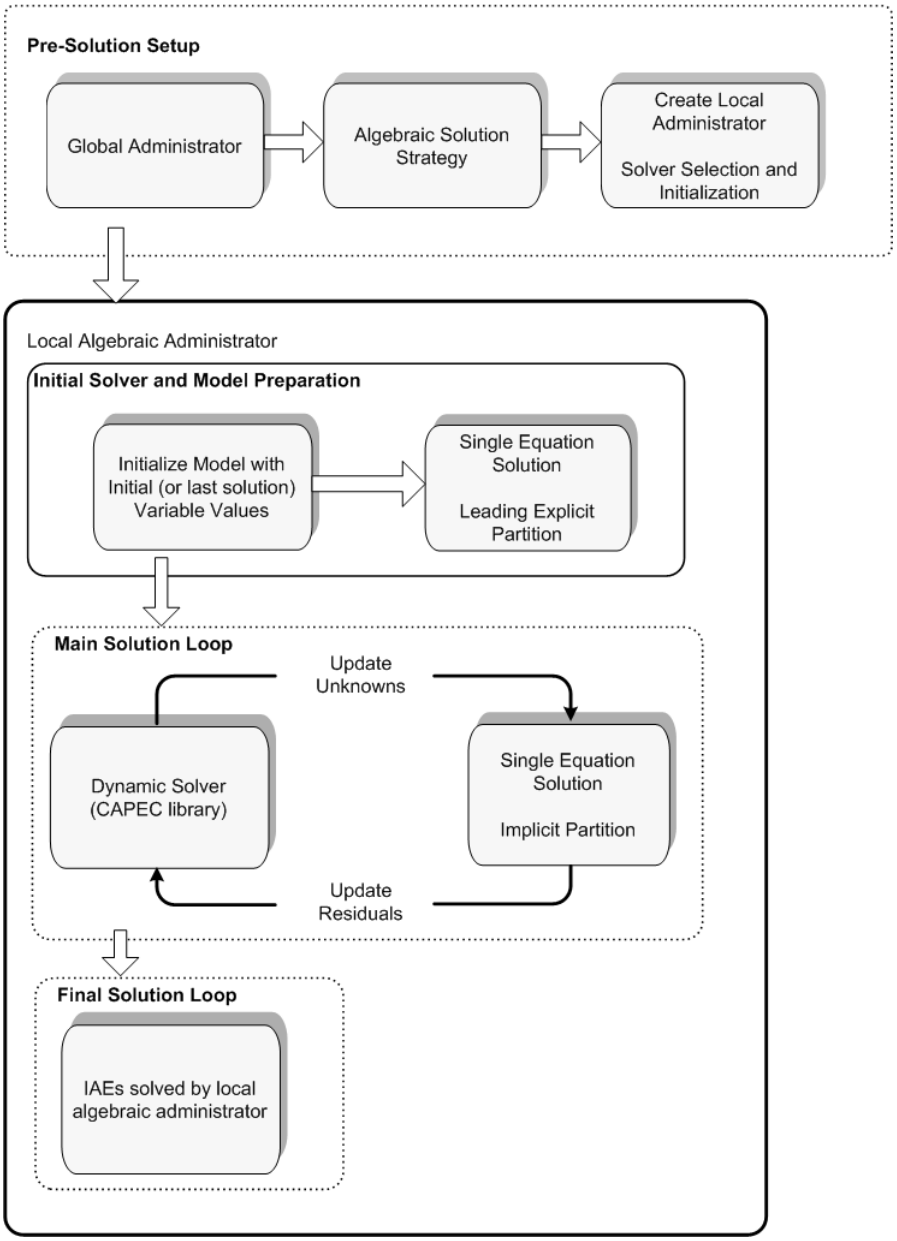


Figure 3.5: Flow diagram for Algebraic solution

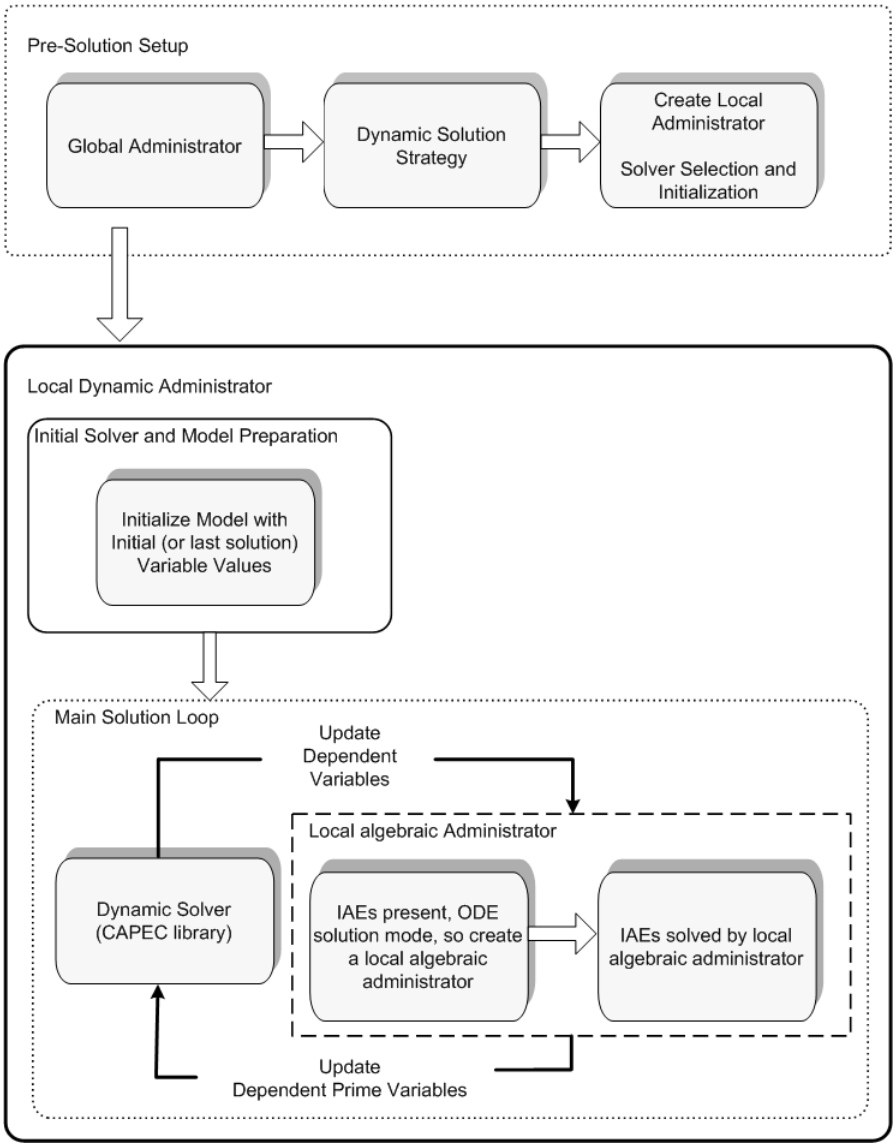


Figure 3.6: Flow diagram for ODE/DAE solution

An example of the interfaces used and links instantiated by the global administrator, the solution of a dynamic optimisation problem is illustrated in Figure 3.7. In this example a local dynamic administrator and local optimisation is instantiated by the global administrator. The dynamic optimisation problem set-up has 3 additional options to specify:

- Manipulated list
- Constraint list
- Measurement list

The list is an array of items to specify, allowing for specification values at a specific time or a specific time interval. The list will be related to a specific variable present in the model, and this variable will be updated accordingly to the list and the current time in the solution.

Figure 3.8 shows the algorithm for solving an algebraic optimisation problem.

What makes this approach flexible and powerful is that from the model classification it automatically detects which of the above local administrators can be applied to a given model, an that the user can choose any of these, e.g. a dynamic optimisation problem with time variant variables can also be solved in a steady state simulation mode by just changing one solver argument.

3.3.5 Model Validation

An important part of the mathematical modelling cycle is validation of the proposed model. Various statistical tests and residual analysis tools are available including marginal and simultaneous tests for parameter significance and correlation analysis of residuals computed from validation data sets, i.e. data sets that were not used for parameter estimation. Appendix B.2 gives a brief discussion about estimation of the covariance and calculation of confidence regions for the parameter estimates that are implemented in the CAMS developed in this project.

3.3.6 Model decomposition

A given phenomenon or unit operation can be represented by different mathematical models or equations and these will differ in their complexity/detail and potential fidelity. For example in a flowsheet simulation, a distillation unit may be represented by a simple component split, by a simplified model such as the Fenske equation or by a detailed plate-to-plate model. These levels may reflect different choices or phenomena, the choice of level between these options clearly has a major influence on the accuracy of the result and on the runtime of the model.

Depending on the purpose for which the model is being assembled, simple, detailed or mixture of levels may be appropriate. For example, a "mixed-granularity" model might be appropriate if the primary purpose of the model

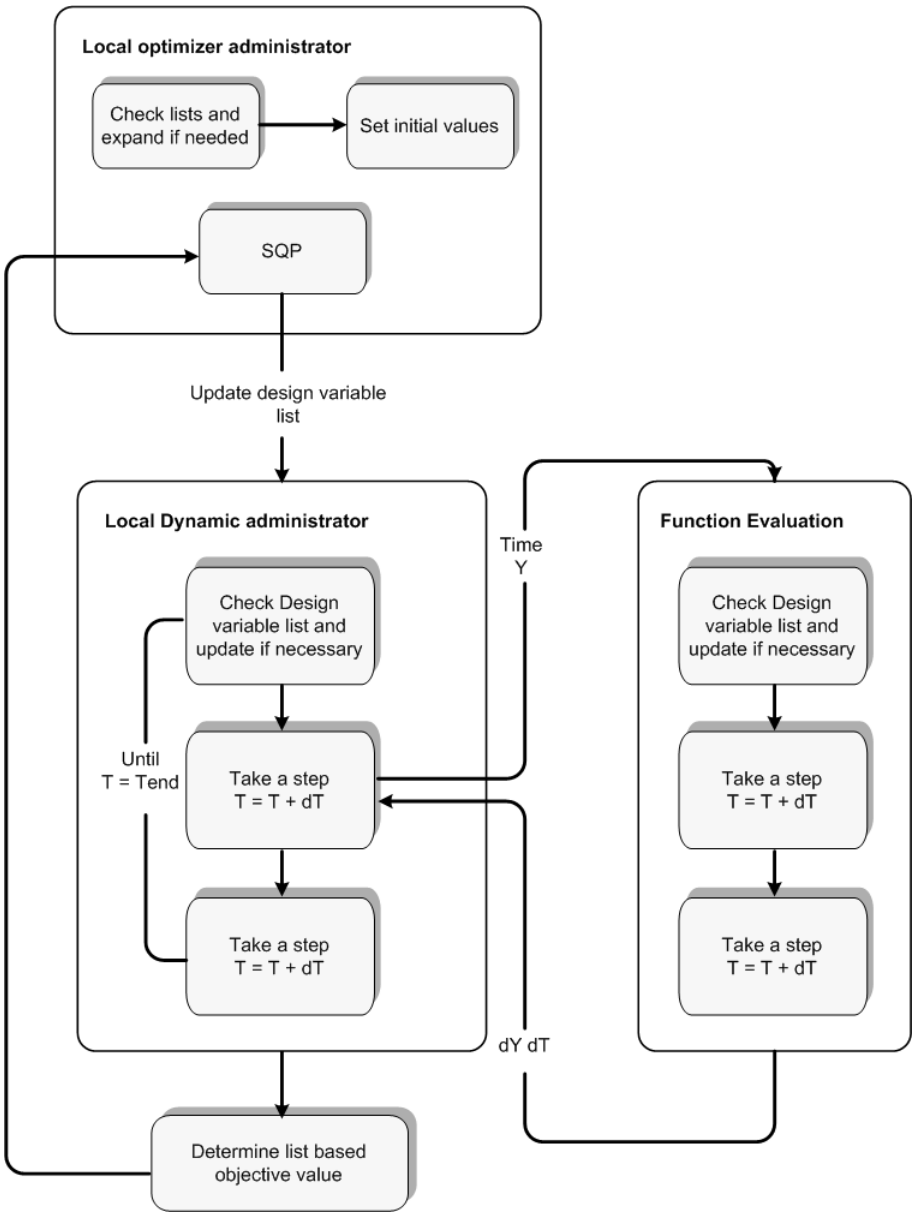


Figure 3.7: Flow diagram for dynamic optimisation

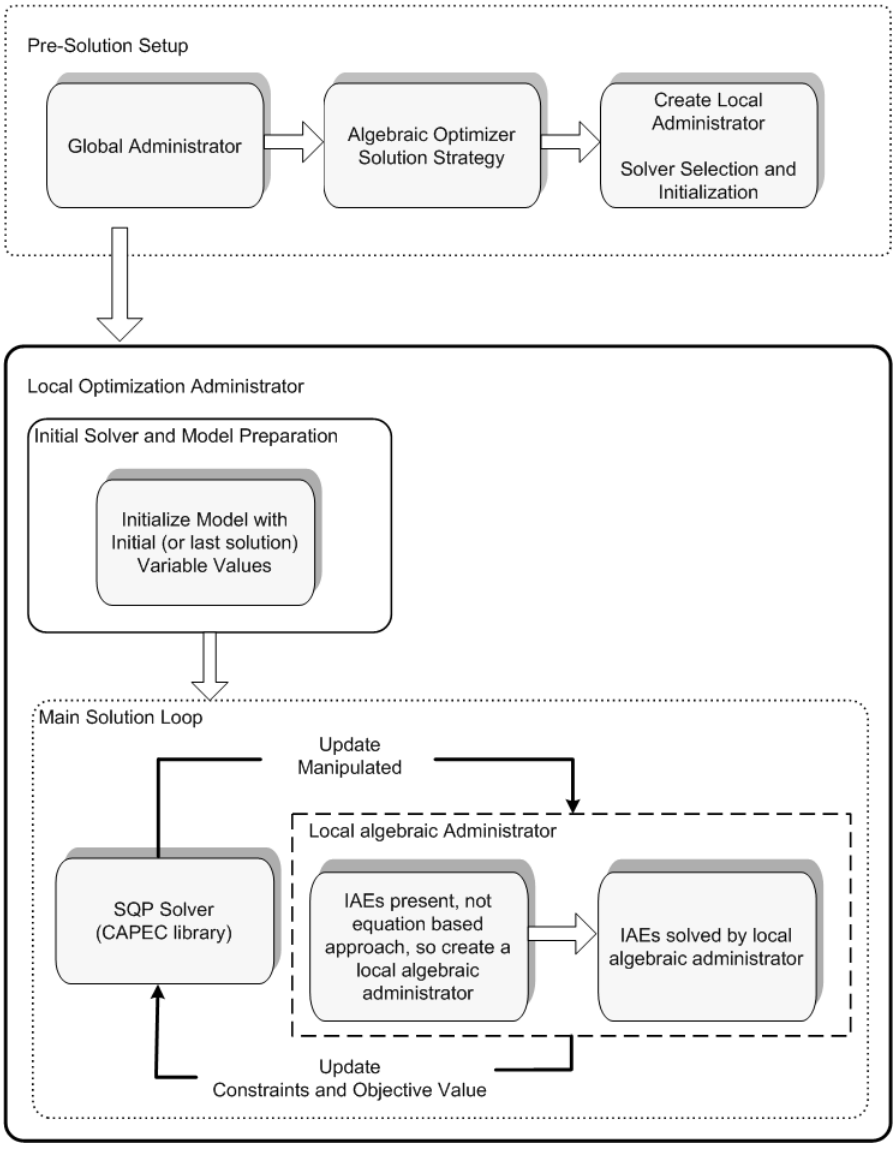


Figure 3.8: Flow diagram for optimisation

is to review the performance of the new reactor (detailed model) within the context of an existing flowsheet (for which simplified models may suffice, at least initially). This model decomposition might have a number of advantages, for example:

- It provides the potential for extreme modularity in a hierarchical structure.
- It provides a rational and consistent basis for model reduction and mixed-granularity models
- The performance and robustness of the simplified models could be predicted and "guaranteed".
- An extreme form of "level" could be a "black-box model"
- It provides a consistent basis for providing inter-working with other major systems.

These concepts have been explored and added to the CAMS developed in this project which is able to perform the calling procedures (external models) allowing the model decomposition.

3.4 Exporting Model

An interface needs to be designed to help the CAMS users work with other application environments. This interface has to support access to dynamic link libraries (DLLs) and servers based upon Component Object Model (COM). This support could allows Visual Fortran (VF), Visual Basic (VB) and Visual C++ (VC++) developers to use the popular mechanisms that make functionality (services) available to other software. Calling code that is written in another programming language is, in general, difficult. There are complex issues around calling standards and data type representations. Solving these problems requires understanding the intricacies of calling standards and single stepping through assembly code. The Fortran and C++ interface deals with the difficulties making COM objects very easy to use from other applications.

3.4.1 COM-Objects

COM-Object is a software architecture developed by Microsoft to build component -based applications. COM is an extensible architecture that provides mechanisms for creating and using software components. A software component consists of reusable pieces of code and data in binary form that can be plugged into other software components from other vendors with relatively little effort. COM objects are more versatile than Win32 DLLs because they are completely language-independent, have built-in interprocess communication capability and easily fit into an object-oriented program design.

The whole point of COM is that the user should not need access to the source code of the COM object that is being used. The very good thing about COM components is that they are never linked to any application. The only thing that an application may know about a COM object is what functions it may or may not support.

3.5 Importing Models

In order to create a simulation, a mathematical model is needed. The possibility to import mathematical models in a native text file format (Equation File) must be available in any CAM system. The mathematical model might be created in a text editor program and then saved as a text file. Then the model will be pre-screened and pre-translated to make sure that it is valid.

The CAM system should offer many new capabilities for users to write mathematical models in different formats, such as XML-files (EXtensible Markup Language). In the real world, computer systems and databases contain data in incompatible formats. One of the most time consuming challenges for model developers has been to exchange data between such systems over the CAM system. Converting the data to XML can greatly reduce this complexity and create data that can be read by different types of application.

3.6 Summary

The objectives of the CAMS developed in this project has been described together with a detailed description of the proposed architecture of the CAMS. Also, a detailed description of the methods and tools needed for the CAMS has been provided. Successful implementation of the architecture and incorporation of the methods and tools should lead to a versatile, robust and flexible modelling system. The next two chapters describe the developed software based on the this architecture (chapter 4) and highlight successful applications (chapter 5).

4

ICAS-MoT: Software for CAMS

The CAMS described in chapter 3 has been implemented into a computer aided modelling tool called ICAS-MoT. The objective of a chemical process modelling tool is to support the steps of any modelling methodology. However, it is not possible or desirable to automate the complete modelling process. Instead, some parts of the modelling process are formally understood, whereas others must be considered as a *creative activity* and therefore require user interaction ((Lohmann and Marquardt 1996)). It is, therefore important to identify those parts of the process, which can be automated in order to formalize and implement them. This chapter describes the main features of the ICAS-MoT as a computer-aided modelling tool while Chapter 5 highlights examples of its applications to various modelling tasks within process/product design.

4.1 Implemented CAMS Features

The functionality of the ICAS-MoT (the modelling tool) has been structured according to the support it provides in the creation and manipulation of mathematical models, in their analysis (e.g. correctness and solvability), and in their translation into an efficient representation suitable for the selected solver.

The main features of ICAS-MoT can be described as follows:

1. Models are coded in terms of sets of variables and equations. Thus models can be expressed in terms of combinations of partial differential, ordinary differential or algebraic equations.

2. Models are entered (imported) as text-files or XML-files, which are then internally translated. Also they can enter directly from the keyboard. No re-coding or complex manipulations are required, the equations may simply be typed in.

3. After an interactive model analysis step (ordering of equations, setting of degrees of freedom, checking of singularity, etc.), the appropriate solver for the model equations is selected together with a solution strategy. As

solver options, ICAS-MoT provides algebraic equations (AEs) solvers, differential-algebraic equations (DAEs) solvers and numerical optimisation methods.

4. ICAS-MoT is able to handle discontinuities in models. This is, equations that take different forms under different conditions are described through use of the conditional sentences (e.g. IF-THEN-ELSE).

5. Matrix manipulation is possible. This feature was included so that it permits users to easily manipulate the rows, columns, sub-matrices and individual elements of a variable given as 2D-Array. Standard matrix operations, such as addition, multiplication and transpose, have been implemented.. This feature is useful in solving systems of equations.

6. The main numerical code in ICAS-MoT handles the solution of large sets of mixed systems of ordinary differential and algebraic equations (DAEs), with facilities for initialization and automatic detection and handling of discontinuities. Partial differential equations (PDEs) are converted automatically to DAEs by discretising all non-temporal dimensions using finite difference approximations (Method of Lines: MoL).

7. Statistical and/or numerical reports for the problem solution and generated. Statistics for the parameter regression step in model identification, in the form of statistics on model solution, the ANOVA report(Analysis of Variance), regression factors and confidence limits.

8. Sensitivity analysis can be formed. The effect of each parameter present in the model, is evaluated by perturbing its value with respect to its reference value and computing the output response in terms of percentage change of the state variables.

9. Dynamic parameter optimisation. This feature allows the solution of optimisation problems that involve models represented by DAE systems, such as model parameter identification when experimental measurements (as function of time) are available.

10. Data bank connection. Data bank access for all compound properties constants and stream definitions have been implemented for flowsheet simulations through the ICAS simulation engine ICASsim or DynSim.

11. Model transfer. Transfer of models represented by AEs, ODEs, DAEs and optimisation to external applications are made through the generation of COM-Objects. The generated, COM-Objects can be used in Excel through a general Excel-COM macro interface, this interface permits loading, manipulation and evaluation of ICAS-MoT models. Through COM-Objects, any MODEL (AE, ODE, DAE) can be used in both ICASsim and DynSim as interconnected modules in a complete flowsheet simulation. Thus, a customized simulator can be created from the models of

the operations in a process flowsheet. ICAS-MoT models can also be used from Visual Basic, Fortran and Visual C++ through their corresponding COM-Object.

12. ICAS-MoT Messages. ICAS-MoT generates text messages to provide assistance during problem solving. These messages informs the user if something is incorrect and indicates how to correct it. All user inputs, equations and data, are checked for format and syntax upon entry, and feedback is immediately. Correct input is required before proceeding to the problem solution.

13. Solution and debug mode.. The solution of a single equation at a time (Equation by equation) is an option to check if either the values passed to the equation are correct or if the derived function is correct. On debug mode the current equation that is being solved is marked and the result will be displayed and the variables value will be updated.

The coarse organization of the ICAS-MoT is shown in Figures 4.1, 4.2 and 4.3, where three major sections can be identified: **Model Definition**, **Model Solution** and **Sensitivity Analysis**.

4.2 Model Definition

This is the starting point for the creation of a mathematical model. This section allows certain manipulation and modification of the model. ICAS-MoT contains "built-in rules" that assist the user in setting up the model equations. Thus, the goal of this feature is to let the user write or retrieve a process model in terms of a set of equations. The model definition process includes:

1. Model creation That is, writing a derived set of model equations according to the syntax rules of ICAS-MoT. The *Create Model* option invokes the new model creation procedure. All model equations must be typed or imported. The order of the equations is not essential at this point, since this can be changed later by the *Variable Analysis* option. After all model equations have been given (i.e. typed, see example in Figure 4.4) the system automatically goes to the *Import Model* option.

2. Import Model. This option invokes the import model method. The name of a file, which holds the model in a text format must be given. The model will be pre-screened and pre-translated to make sure that it is valid. If a new model is imported, any existing model (including classification and variable settings) will be discarded, and the imported model will be used instead. The model can be modified afterwards by the *Modify Model* option.

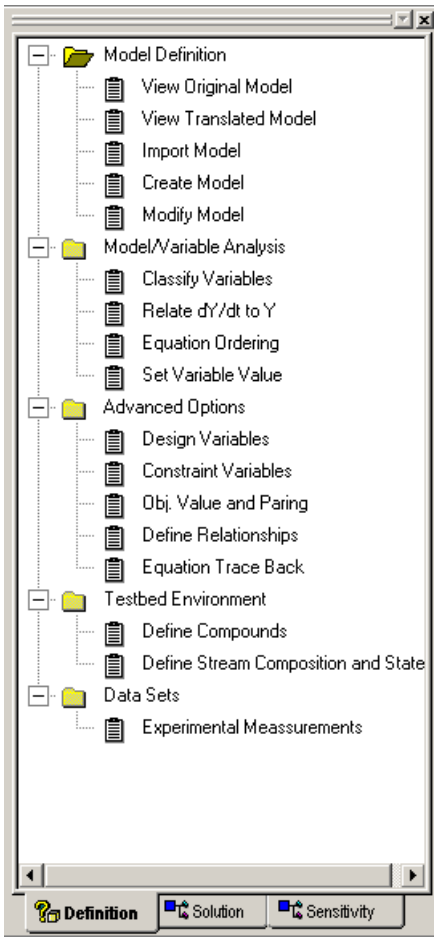


Figure 4.1: Problem definition and model/variable analysis sections.

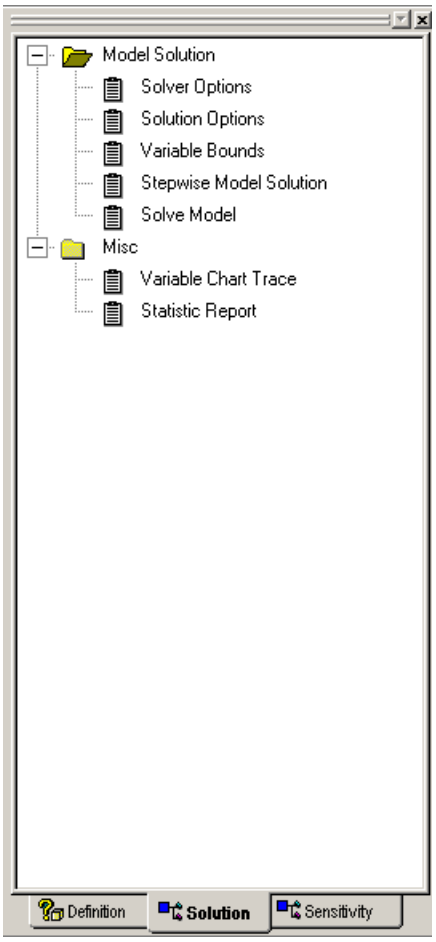


Figure 4.2: Problem solution section.

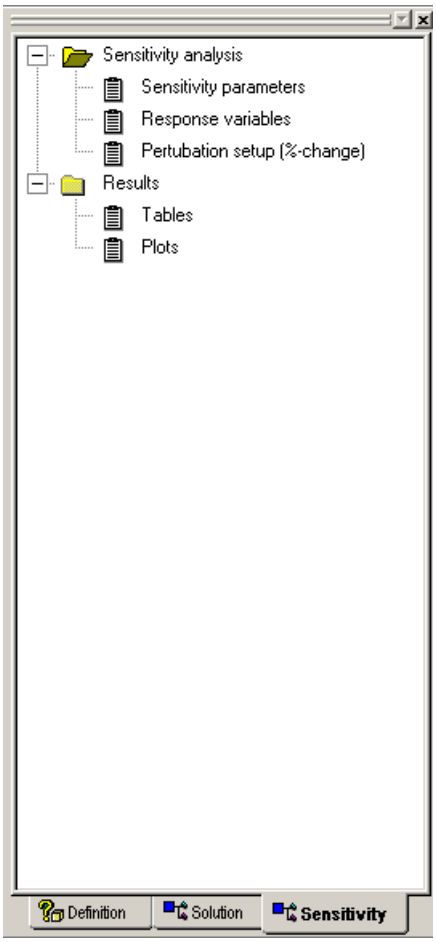


Figure 4.3: Sensitivity analysis section.

```
#Polymerization Reactor

kp  = Ap*exp (-Ep/R/T)
kfm = Afm*exp(-Efm/R/T)
kI  = AI*exp (-EI/R/T)
ktd = Atd*exp(-Etd/R/T)
ktc = Atc*exp(-Etc/R/T)

P0 = sqrt(2.0*fe*CI*kI/(ktd+ktc))

#Dynamical Model
dCm = -(kp+kfm)*Cm*P0+F*(Cmin-Cm)/V
dCI = -kI*CI+ (FI*CIin-F*CI)/V
dT  = mDH*kp*Cm*P0/(ro*Cp) - U*A*(T-Tj)/(ro*Cp*V)+F*(Tin-T)/V
dD0 = -(0.5*ktc-ktd)*P0*P0 + kfm*Cm*P0 - F*D0/V
dD1 = Mm*(kp+kfm)*Cm*P0-F*D1/V
DTj = Fcw*(Tw0-Tj)/V0 + U*A*(T-Tj)/(row*Cpw*V0)

# % monomer conversion
X=(Cmin-Cm)/Cmin *100

#Average molecular weight
PM =D1/D0
```

Figure 4.4: ICAS-MoT model creation step (source code)

3. Modify Model. This option allows edition and modification of the equations in the model. Equations and variables can be added or removed from here. If the model is changed it will be re-screened and re-translated but the variable information (classification, type and value) will be preserved for existing variables. After the model modifications have taken place, a list-box will display the variables that were removed.

4. View Original Model. This option displays the original model as it was typed (before translation). Any comments present have been preserved.

5. View Translated Model. This option displays the results of the translation. The Model will be displayed as a tree view, where the original equation is the node and the expanded equation(s) is (are) the translated equation(s). A section of a translate model is shown in Figure 4.5.

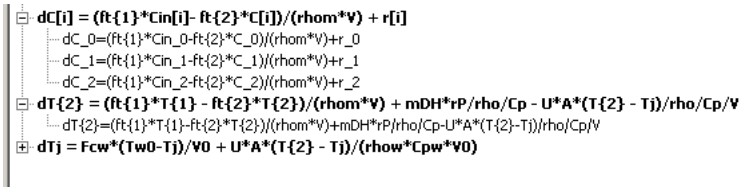


Figure 4.5: Model equation expanded

The important step here is the translation that dissects the text-based equations using a Reverse Polish Notation (RPN) algorithm and classifies equations

and variables using a multi-layered classification system for equations and variables. First, ICAS-MoT identifies the equations and variables according to a set of simple syntax rules in the equation editor:

1. The editor is case sensitive.
2. Comments can be added by using the symbol "#" or ";" preceding the comment.
3. There is not a reserved name for time and position variables, but the user may use, t or x, respectively to represent them.
4. Equations must contain and equal sign ("=").
5. In general, any variables on the right hand side (RHS) must be classified in the corresponding dialogue boxes, as well as names of dependent variables must be related (defined) in the corresponding dialogue boxes.
6. The ODEs or PDEs should be arranged so that time derivatives appear on the left hand side (LHS) of the equation.
7. The derivative term must start with a "d" or "partial" to indicate an ODE or a PDE, respectively.
8. If more than one operator is placed between two variables, only the first one will be considered.
9. Variable names must not have any operator.
10. Only one variable or a derivative operator is allowed on the LHS of any equation.
11. An equation with a zero on the LHS is counted as an implicit AE.

Based on the above syntax rules, ICAS-MoT identifies the equations and classifies (according to the first-translation layer), as:

- Algebraic Equations (AEs)
- Implicit Equations (having more than one unknown variable per equation)
- Explicit Equations (having only one unknown variable per equation)
- Ordinary Differential Equations (ODEs)
- Partial Differential Equations (PDEs)

The variables are classified (in this first-translation layer) in terms of those appearing on the LHS and RHS of the equations. All variables listed as appearing on the RHS of equations are classified (by default) as parameters.

The translation algorithm scans the original text based equations and checks for mathematical consistency (the number of equations and variables before translation must be the same afterwards). The syntax rules listed above are used to interpret the text based model equations. If the model passes this validation test, each equation and variable is expanded through the RPN algorithm. This means that a large number of expanded equations and variables need not be specified as the initial model, making the import of external models more flexible and easy. Once the equations have been expanded, the translation step is completed.

4.3 Model/Variable Analysis

The model resulting from the modelling process is often not well suited to be directly used by the solver. A number of simplifications and optimisations are usually applied to the model as it is transformed. For example, a discretisation based on the method of lines (MoL) approach is employed to make PDAE systems solvable by standard DAE solvers (Pfeiffer and Marquardt (1996)). Optimisations of the equation structure can be performed, e.g. by reordering equations, eliminating explicit equations, or factoring out common equation structures or sub-expressions (Allan and Westerberg (1999); Morton and Collingwood (1998)). The final step of a modelling tool is to transform such a declarative model into a procedural representation to be used by the solver.

4.3.1 Variable Analysis.

1. Classify variables. This analysis step invokes the second-analysis layer for the classification of variables (Figure 4.6). In this layer, all variables identified previously as those appearing on the RHS of equations and classified automatically as parameters, are now re-classified by the user as:

   (a) Parameter: variables with known values

   (b) Explicit: variables that are function only of parameters and/or dependent-prime variables

   (c) Implicit-Unknown: variables related to AEs where there are more than one unknown variable per equation

   (d) Dependent: variables appearing with the differential operators on the LHSs of ODEs and/or PDEs

   (e) Dependent-prime: the derivative operator related to the dependent variable)

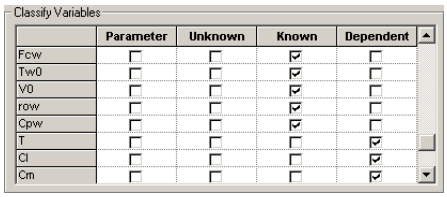


Figure 4.6: Variable classification in ICAS-MoT

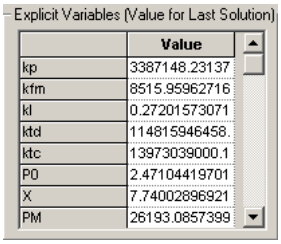


Figure 4.7: Explicit Variables automatic classified

Any changes in the variable classification will immediately be reflected in the incidence matrix. Figure 4.7 shows the detected explicit variables and the last calcualted value (default is zero). If no solution has been made for the model, all variables will be set to zero by default. Note that according to this classification, the LHS variables can only be Unknown, Dependent and/or Dependent-Prime. Once the classification of the variables according to the rules of the analysis layer has been made, ICAS-MoT offers a number of analysis options:

- Generation/analysis of incidence matrix (with equations as row index and variables as column index). The incidence matrix shows a visual interpretation of the ordering of equations and classification of variables (Figure 4.8). The matrix is flexible so the user can choose whether or not to show the colour-coding, the parameters or the known variables. If a different ordering of the equations is desired, the user can move a single equation up and down or switch two equation locations. In this way changes in the equation ordering or variable classification is immediately shown in the incidence matrix.
- Check for singularity of matrix (identification of equations with no unknown variables). If the model contains differential equations and unknown variables, a singularity check will automatically be performed when the *Classify Variables* window is displayed. If a singularity is found, a message box will appear with the variables, that causes the singularity.

2. Relate dY/dt to Y: This option is only for models with differential equations and allows to the user to relate the dependent variable (Y) with the appropriate differential variable (dY/dt). Before using this option, one or more variables (depending on the model) must be classified as *Dependent variables* in the "Classify Variables" node. The user will only be allowed to choose a "differential prime" variable among the Explicit variables (Figure 4.9).

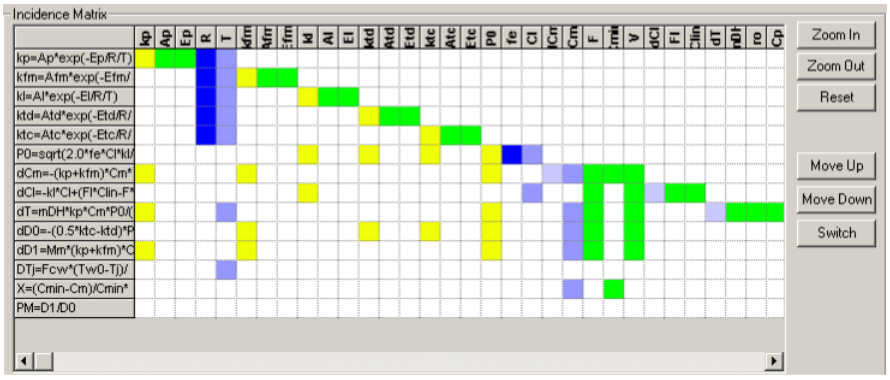


Figure 4.8: Incidence Matrix or occurrence matrix

Dependent Variable	Dependent Prime
T	dT
Cl	dCl
Cm	dCm
Tj	DTj
D0	dD0
D1	dD1

Figure 4.9: Paring differential variables

3. Equation Ordering. Decomposition, partitioning and ordering of the model equations are shown. This feature identifies the sub-sets of equations that need to be solved simultaneously and finds the best equation order evaluation. This will ensure a robust and efficient solution, hence, equation ordering is recommended for all models to be solved in ICAS-MoT (Figure 4.10).

4. Set Variable Value. This option allows the specification of the variable values used in the model. The values specified here will override any previously specified values. Guesses, initial states, parameters and known variables must be specified here.

Furthermore a degrees of freedom analysis is automatically performed to ensure that the problem is not ill-posed (e.g. number of equations does not match the number of unknown variables) before going to the solution step.

4.3.2 Advanced Options

1. Design Variables. This option allows the definition of variables to be used in the optimiser. The variables to choose are from the "Parameter" or "Known"' variable class, because the value of these variables can be changed during the optimisation procedure.

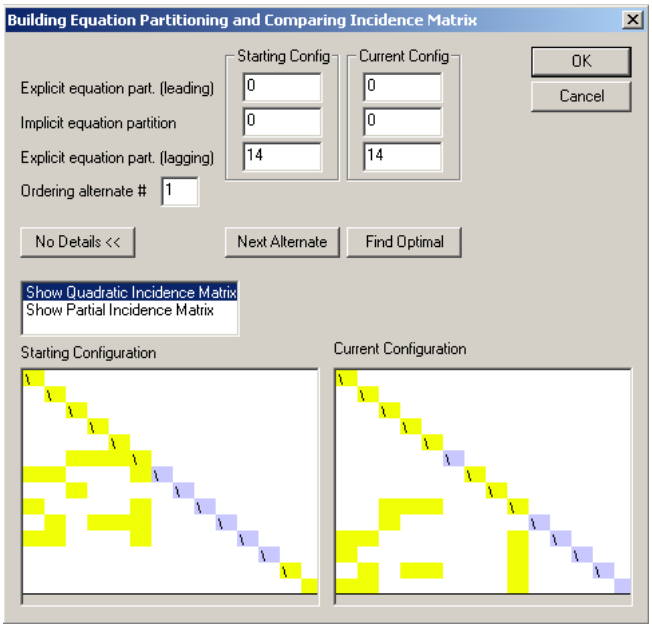


Figure 4.10: Equation partitioning and incidence matrix comparison

	Lower Bound	Design Var	Upper Bound	Initial Value	Final Value	Active
1	0	k1	20	10	10	☑
2	0	k3	20	1	1	☑
3	0	k2	20	15	15	☑

Figure 4.11: Design variables

2. Constraint Variables. This option allows to select the variables to be used in the optimiser. The variables to choose from here are from the "Explicit variables" classes. The lower bound, upper bound and initial value of each design variable selected (i.e. active) must be defined here.

3. Obj. Value and Paring. This option allows the selection of variables from the "Explicit variables" classification (LHS) that will be considered as the objective function variable (Figure 4.12). The variable selected must be a single variable, and it cannot be a data collection (array variable). This section also allows the selection of a differential variable that should be updated during integration or when the steady state is achieved.

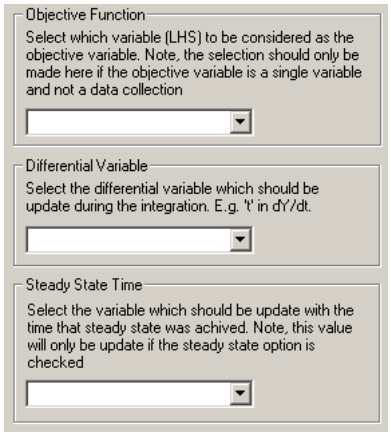


Figure 4.12: Custom variables relationship

4. Define relationships. This option displays the relationships defined in the model. A relationship links special model variables to the model solution and the simulation engine (e.g. a default relationship links the pressure in a stream to the variable "P"). The relationship can also be used to define iterative variables. The number of compounds present have the default relationship 'nc' as the number of compounds, and 'i' is the index in the model equations (the number of compounds selected gives the maximum value of 'i'.) (Figure 4.13).

	Formal Name	Iterative Name	Starting value	No Elements	Relates To...	Notes
1	nc	i	0	0	Number of Compounds	Relates the counter 'i' to the maximum number of compounds
2	nr	m	0	1	Number of Reactions	Relates the counter 'm' to the maximum number of reactions
3	ft	<NONE>	0	0	Total Stream Flow	Informs the engine that 'ft' is total flow of a stream
4	f	<NONE>	0	0	Comp. Stream Flow	Informs the engine that 'f' is the compound flow in a stream
5	T	<NONE>	0	0	Stream Temperature	Informs the engine that 'T' is the stream temperature
6	P	<NONE>	0	0	Stream Pressure	Informs the engine that 'P' is the stream pressure
7	ND	n	0	10	Number of Discretization point	Informs the engine that 'n' is the number of Discretization point
8	X	x	0	1	Spatial domain x direction	'x' is the spatial domain of Discretization points in PDEs
9	k	k	1	5	Index	Index relationship

Figure 4.13: Relationship definition

4.3.3 Test-bed Environment Section

- Define Compounds. This option displays the dialog box for selecting the compounds to use in the model. The user can either choose from predefined sets of compounds or create a new set (Figure 4.14). If the model contains any equations that are related to the number of compounds then *define compounds* must be specified before a successful translation can take place (usual relationship relates the counter 'i' to the number of compounds).

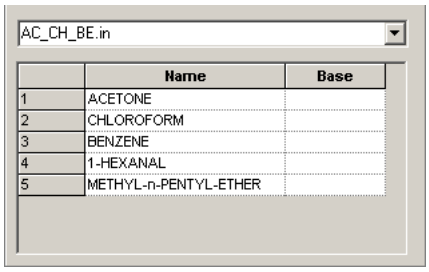


Figure 4.14: Compound selection

- Define Streams. This option displays the dialog-box for specifying the streams present in the model (Figure 4.15). This option is used to specify the pressure, temperature, and compound flows. Only the variables used in the model will be available (e.g. if the pressure is unused by the model, it will not be shown). The variable type will also be displayed —It is, thus, possible to recognize if the variable values will be used for true values or initial guesses. This option can only be used if relationships that relate model variables to pressure, temperature and composition to the simulation engine have been defined.

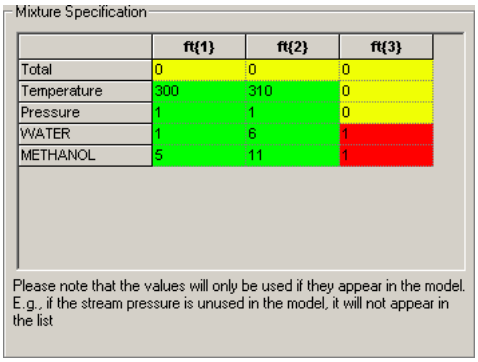


Figure 4.15: Stream definition

4.3.4 Data Set

- Experimental Measurements. This option displays a dialog-box that allows the introduction of experimental measurements and to relate them with the variables present in the model (Figure 4.16). Dynamic kinetic parameter optimisation is related to this option.

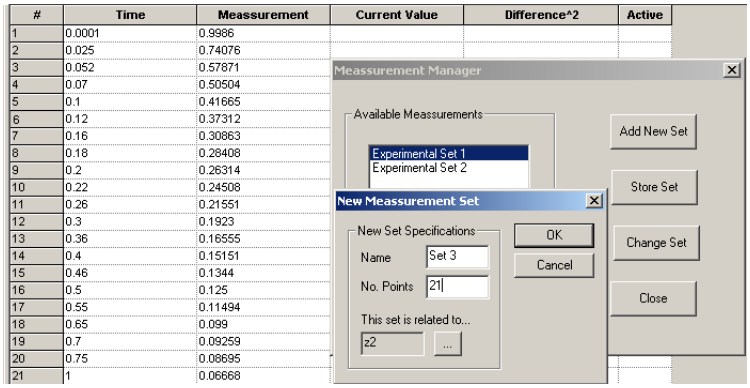


Figure 4.16: Experimental measurements input.

4.4 Model Solution

All of the model formulations presented above need to be assigned a solution algorithm to actually find a solution from some given input, such as, the parameter values for an algebraic equation system or the initial conditions in the case of a differential algebraic equation system. Without the solver algorithm, the model as formulated above can only be used to check whether a given state is among the possible states of the system under consideration(functions evaluation only). This functionality of the model is used by the solver to iteratively determine a solution of the model.

The solution step in ICAS-MoT involves the following procedures:

- Administrator: Administration of the solution procedure (drives the numerical solution task).
- Solver-link: Connection to the solver specified by Administrator.
- Residuals: Computation of the function values (RHS) for the translated equations.

The *Administrator* is divided into a global administrator and one or more local administrator(s), as needed by the solution strategy. Each local administrator instantiates one or more solvers from the solver library. The possible local administrators are (Figure 4.17):

- Algebraic administrator
- Integration administrator
- Optimisation administrator

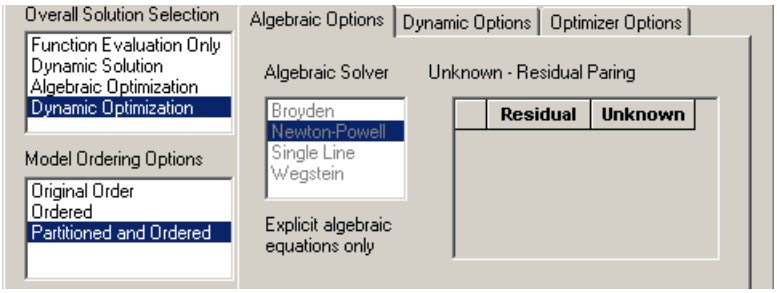


Figure 4.17: Model solutions mode

The global administrator combines all the local administrators needed to solve a given problem and handles the overall solution sequence. The local administrators are only connected to the model via the global administrator. From the model classification information, ICAS-MoT detects and lists the needed local administrators (see Figure 3.4, where the structure of the local administrators is highlighted). If more than one local administrator is needed, the variable classification rules for the third-solution layer are invoked. Here, variables are classified as:

- Design/Manipulated: parameters whose values may be changed in the outer-loop.
- Constraints: unknown, dependent and/or dependent-prime whose values must be within some specified bounds.

If the problem solution also involves parameter estimation based on supplied data, the data variables need to be classified as Real, Integer and Binary. Once the variables have been identified, the appropriate solver linked to ICAS-MoT is invoked through the global administrator. Note that in problems related to optimisation (parameter estimation, process optimisation, etc.), multiple solvers may be used if the process model equations are solved separately from the objective function and constraints (Figure 3.7).

- Solver Options. This section displays the special options and settings for the solvers present in ICAS-MoT. A predefined "Default Settings" is available, but the user can customise the solver setting as well.
- Solution Options. This section displays the options to be used when the model is solved. The selection of the default solver to be used can be

done from here. If a variable trace is needed (variables plotted during solution), an option 'Allow variable Trace' is available. Bear in mind that when variable trace is enabled, the solution will be slower since it will take time to update the graphs.

- Variable Bounds. This section will display the current bounds for the unknown variables. Default bounds are Lower = 0 and Upper = 1000.
- Debug Mode. This section will display the translated model equations and allows the solution of a single equation at the time (Figure 4.18). After each equation has been solved, the result will be displayed and the variables will be updated. Equation-by-equation solution mode is an equation debug option to check if either the values passed to the equation are correct or if the derived function is correct.

	Break	Equation	Result
1	☐	kp=Ap*exp(-Ep/R/T)	29701
2	☐	kfm=Afm*exp(-Efm/R/T)	3.55255e-005
3	☐	kl=Al*exp(-El/R/T)	8.83854e-016
4	☐	ktd=Atd*exp(-Etd/R/T)	5.3548e+010
5	☐	ktc=Atc*exp(-Etc/R/T)	6.51676e+009
6	☐	P0=sqrt(2.0*fe*Cl*kl/(ktd+ktc))	9.23835e-014
7	☐	dCm=-(kp+kfm)*Cm*P0+F*(Cmin-Cm)/V	64.678
8	☐	dCl=-kl*Cl+(Fl*Clin-F*Cl)/V	-4.744
9	☐	dT=mDH*kp*Cm*P0/(ro*Cp)-U*A*(T-Tj)/(ro*Cp*V)+F*(Tin-T)/V	-162.818
10	☐	dD0=-(0.5*ktc-ktd)*P0*P0+kfm*Cm*P0-F*D0/V	4.29207e-016
11	☐	dD1=Mm*(kp+kfm)*Cm*P0-F*D1/V	0
12	☐	DTj=Fcw*(Tw0-Tj)/V0+U*A*(T-Tj)/(row*Cpw*V0)	5756.58
13	☐	X=(Cmin-Cm)/Cmin*100	100
14	☐	PM=D1/D0	0

Figure 4.18: Debug mode solution

- Solve Model. This option starts the solution of the model (Figure 4.19). If the model is a differential model, there is an option in "solve model" that can evaluate the functions only. This will use the algebraic solver, and it does not do any integration.

4.4.1 Miscellaneous Section

- Variable chart trace. This section is used to choose the variables that must be plotted during the model solution process.
- Statistics report. In this section a statistics report (see Figures 4.20 and 4.21) including (among other information) the model metrics, correlation factors and confidence intervals, is generated by ICAS-MoT after the model solution.

4.5 Sensitivity Analysis

ICAS-MoT provides a sensitivity analysis option to determine the sensitivity of response variables of the model (or model outputs) due to changes in model

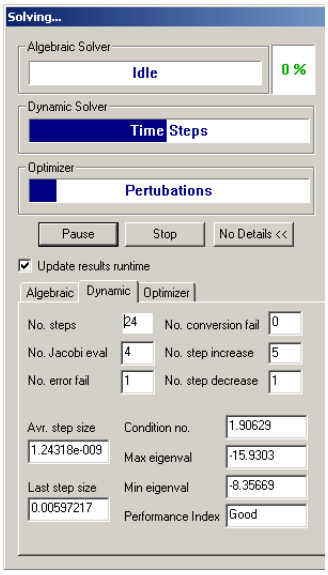


Figure 4.19: Solving model -information

Confidential Interval of the estimated parameters

Name	Value	(+/-)95% CI
k1	1.200377e+001	6.212964e-00
k3	1.996487e+000	1.033350e-00
k2	8.001468e+000	4.141435e-00

Solution Overview:

Number of parameters:	3
Number of known variables:	0
Number of unknown variables:	0
Number of Dependent variables:	2
Number of Dependent prime:	2
Number of explicit variables:	0
Number of design variables:	3
Total number of variables:	7
Total number of equations:	2

Figure 4.20: Statistics report of model and its solution

Sum of Squares

Total sum of square for y (SST y):	0.243152
Total sum of square for y^ (SST y^) :	0.243269
Error (or residual) sum of square (SSE):	2.700292e-008
Regresion sum of squares (SSR):	0.243269

Variance

Sample variance (sy) :	0.012158
Sample variance (sy^) :	0.012163
Sample covariance (syy^):	0.012161

Standard deviations

Sigma y :	0.110261
Sigma y^ :	0.110288
Sigma yy^:	0.110275

Statistics for the regression

Pearson's or Correlation coefficient (R):	1.000000
Coefficient of Correlation or R-Square:	1.000000
Adjusted R-Square:	1.000000
Standard error of the estimates:	0.000038
Observations:	21

Figure 4.21: Statistics report of the estimated parameter

parameters or input (design) variables. This is an important method for checking the quality of a given model (including the robustness and reliability) as well as identifying the most important (sensitive) design variables.

ICAS-MoT performs this analysis by means of systematic perturbations that involve changing the value of one or more selected variables and calculating the resulting change in the unknown (output) variables. Changes in parameter variables can be assessed one at a time to identify the responses of key (output) variables. If a small change in a parameter results in relatively large change in the response variable, the response variable is said to be sensitive to that parameter. This may mean that the parameter has to be determined very accurately or that the process/operation has to be redesigned for lower sensitivity.

The sensitivity analysis involves the definition of:

1. Sensitivity parameters. This option allows setting the variables that will be used in the sensitivity analysis as perturbation variables. The variables to choose are from the 'Parameter' or 'Known' variables classes.

2. Response variables. This option allows to setting which variables must be tested for the sensitivity response. The variables can be chosen from the 'Explicit', 'Dependent' or 'Unknown' variables classification.

3. Perturbation Set up. This section displays the dialog-box that allows to set up the perturbation steps and the upper and lower limits (in percent) for the perturbed variables.

The results are given in terms of:

- Tables. For all the variables selected as "response variables", this section displays its corresponding value at different values of the perturbed parameter chosen.

- Plots. This section displays a graphical view of the sensitivity analysis.

4.6 Model Transfer

After the model equations have been successfully solved, the user has the option to generate a COM-Object of the model to transfer it to the ICAS model library (for use through the ICAS simulation engine), or to use in external software. For repeated use of the model, a model transfer is recommended. Note, however, that if the model equations are changed, the COM-object would need to be generated again. The same COM-object can, however, be used for different sets of parameters, for example, different sets of compound properties, reaction kinetics and equipment sizing data. Several functions have been incorporated allowing the complete control of the model.

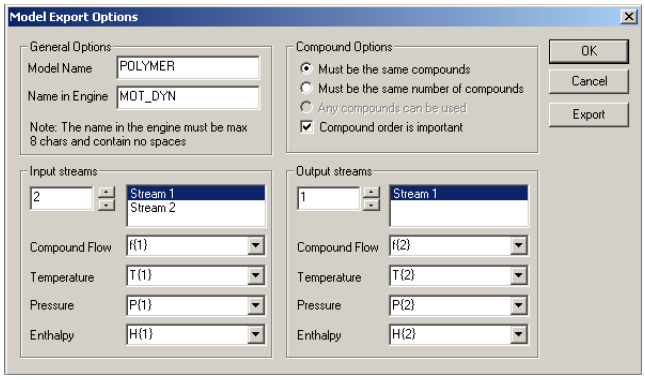


Figure 4.22: Model export interface

- Model exported in ICASsim/DynSim. A model created in ICAS-MoT can be used as a part of a flowsheet and be simulated in both ICASsim (steady state) and Dynsim (dynamic) simulators. The model(s) to be used in the flowsheet should be created and exported (Figure 4.23).

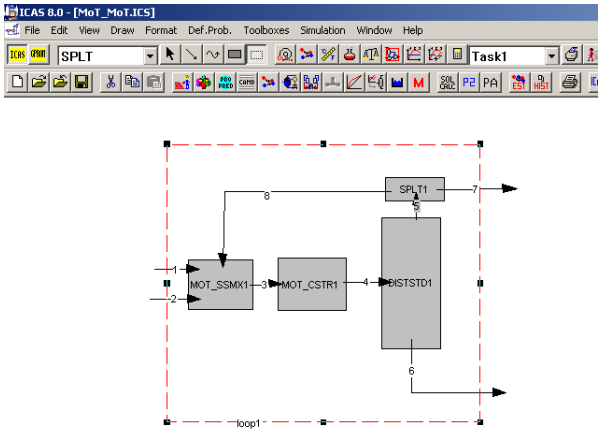


Figure 4.23: ICAS-MoT models imported and used in ICASsim/DynSim

- ICAS-MoT models within Excel. Microsoft Excel is an extremely powerful tool, which is used by millions of people everyday. Functions tailored to a specific task can be programmed into Excel to extend its capabilities with customised analysis tools. A simple method of customizing Excel is to create a Macro. An Excel Macro file that through COM technology is able to use ICAS-MoT models has been developed. The Excel Macro is customized such that it is able to execute a sequence of modelling related

activities: reading of input data, execution of the ICAS-MoT model, and writing of output data (results). In this way, the process of evaluation and use of an ICAS-MoT exported model is completely automatic. The main advantage is that users can prepare data for an ICAS-MoT exported model, get output results from the ICAS-MoT exported model, and store results from the same model using different parameters, directly through an Excel worksheet environment. This makes the model use easier for those not familiar with the modelling tool-box environment.

- Working with other languages. All ICAS-MoT models can be used from Fortran, Visual Basic and Visual C++. The main goal is that ICAS-MoT models can be (re)-used by anyone who has to write programs in these languages and needs a quick way to access some calculation procedure previously developed in ICAS-MoT. This interface allows the access of the COM-object directly from the source code.

4.7 Model Integration

The model equation of the whole process can be deduced by an aggregation of the model equation systems of each process part. The aggregation process should follow the hierarchical representation structure introduced during process abstraction (Section 3.3.6). This not only leads to mathematical models of the partial processes introduced on the various hierarchical levels to facilitate model implementation and reuse, but it also allows a controlled specification of constraints which may result from aggregation. In ICAS-MoT the models can be decomposed into sub-models and used as a sub-model (piece of code) that can be called from a main model, called as part of a nested model (such as kinetic model in a reactor model), and called at several locations.

The VT-flash calculation is used here as example to highlight how ICAS-MoT models can be used as procedures (sub-models).The VT-flash is a common problem in chemical engineering where vapour and liquid compositions have to be calculated for known temperature and available volume (Figure 4.24).

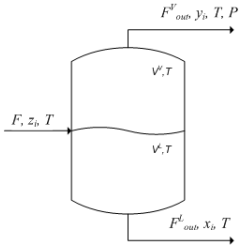


Figure 4.24: Illustration of the VT-flash.

In general, standard PT-flash calculations are performed in the inside loop and the volume constraint is checked in the outer loop, which verifies the pressure value for the PT-flash calculation. In this way the calculation task follows the algorithm presented in Figure 4.25.

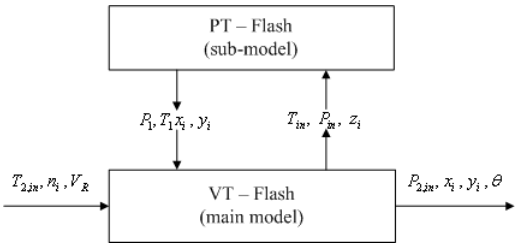


Figure 4.25: VT-flash calculation using a PT-flash as a sub-model

At each iteration the VT-flash ICAS-MoT "main module"' calls a external procedure (ICAS-MoT file) where PT-flash calculations are done. The model equations for this problem are given in the flow diagram presented in Figure 4.26.

The model implementation is given in Appendix D. The results are given in the Table 4.1, where a binary mixture of water($H2O$)- Methanol($MeOH$) is used as a simple test example.

Table 4.1: VT-flash calculated results

Given			Results		
T	340	K	x_{H2O}	0.323	-
V	50	m^3	x_{MeOH}	0.677	-
n_{H2O}	3	$kmol$	y_{H2O}	0.192	-
n_{MeOH}	7	$kmol$	y_{MeOH}	0.808	-
			θ	0.174	-
			P_{in}	0.97	atm

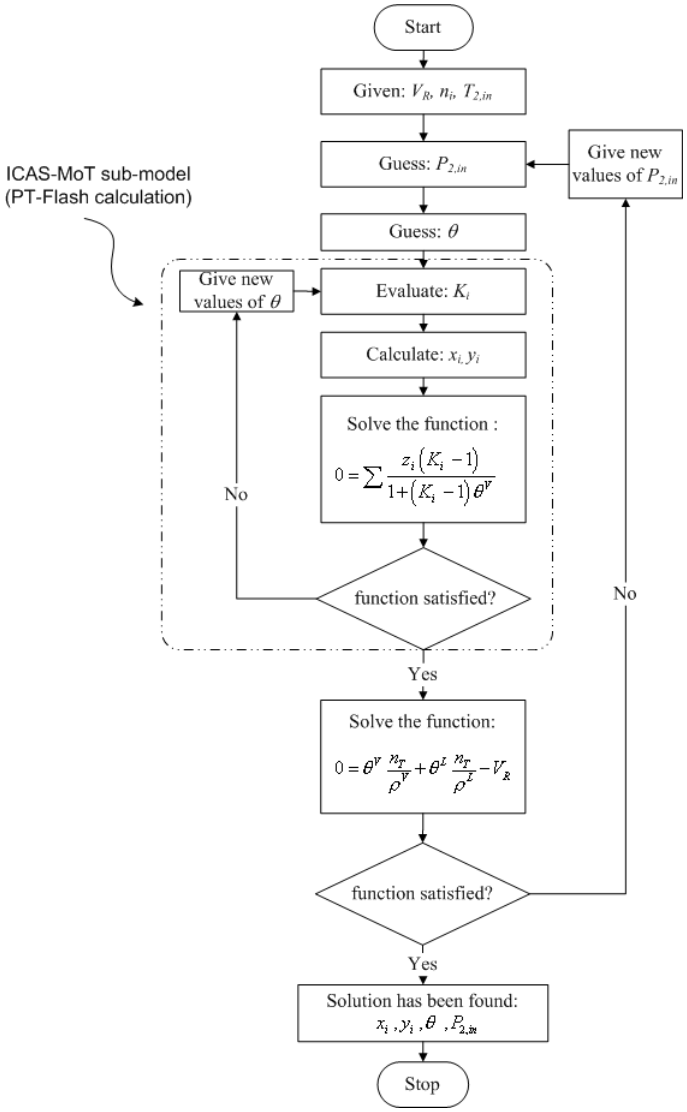


Figure 4.26: Algorithm for solving the VT-flash problem (main ICAS-MoT model)

4.8 ICAS-MoT Software Specifications

In the previous section we discussed the major components of ICAS-MoT. This section describes the software specifications of the developed modelling system (ICAS-MoT). ICAS-MoT employs a flexible equation-oriented approach. It has been designed to deal effectively with a much wider range of processes, including those with combined discrete and continuous characteristics, as well as, lumped and distributed parameter systems. Furthermore, it can perform dynamic optimisation, sensitivity analysis and generate statistical reports.

ICAS-MoT is a windows based application. It forms part of ICAS's toolbox (Gani, Hytoft, Jaksland and Jensen 1997) and it has been designed with the objective of minimizing the amount of effort that it takes to specify, solve, and visualize the solution of a system of Algebraic (AE), Ordinary (ODE, DAE) and Partial differential (PDE, PDAE) equations without sacrificing computational power and flexibility. The CAMS Graphical User Interface (GUI) helps with the navigation of options and commands by following standard Windows conventions. The CAMS input language closely resembles the mathematical notation used to write any system of differential equations (AE, ODE, DAE, PDE, PDAE) and relieves the user of the burden of having to work with arrays, indexes, external subroutine calls, solvers, and many more. Both modules, the graphic user interface and the database access were implemented using VC++ an object oriented programming language, whereas the numerical part that forms the core of ICAS-MoT (thermodynamic and solver libraries) has been implemented in FORTRAN.

In addition ICAS-MoT provides a self-contained computing environment for the solution of systems of partial differential equations (1-D systems). After specifying the system of equations, the CAMS is ready to solve it. The method of solution is based on the numerical Method of Lines (MoL), where spatial derivatives are approximated by finite differences and the resulting set of ODEs is integrated with robust integrators such as BDF and RKF. It has incorporated several algebraic solvers, numerical integrators and optimisers as well, which are easily accessed and thus the entire model solution can be handled in the same environment for a wide range of problems.

ICAS-MoT generates the code for the model and adds it to the model library in ICASsim or DynSim (Gani et al. 1997) so that it is available for all other tools and applications that may be needed. The advantage of the combination of the ICAS-MoT and ICASsim or DynSim is that different models and/or process configurations can be simulated very easily and quickly, once the model has been validated and integrated (automatically) into ICASsim model library.

It is important to point out that all of these mathematical manipulations are carried out in a manner totally transparent to the user who is therefore left free to concentrate on the physics of the problem.

Figure 4.27 shows all the ICAS-MOT options available that can be chosen depending on the type of modelling problem that has to be solved. The options

include several tools to handle and solve a wide range of problem formulations involving AEs, ODEs, DAEs , PDAEs and optimisation problems. For each of these problems some options are activated and others are not activated. Regarding the model export options, two different modes are available: (a) as a COM-Object that can be used in external applications like Excel, VC++, VB, or Fortran, and (b) as an ICASsim unit process to be incorporated into the ICASsim unit library and used to customise a simulator in steady state mode through ICASSim or in dynamic mode through DynSim.

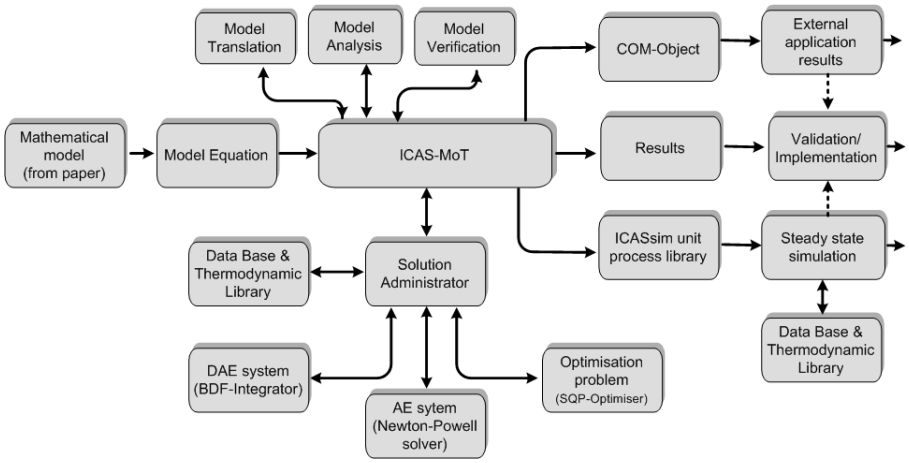


Figure 4.27: ICAS-MoT available options

The work-flow shown in Figure 4.27 highlights all the possible (available) paths to be followed in order to solve any mathematical model. It involves the main steps to set up the model such as model translation, model analysis, model verification, and model solution. But only some options must be followed to solve the specific problem under study, as it will be seen in the short examples and case studies that will be presented in the following chapter.

4.9 Summary

General purpose CAMS have now reached a high degree of sophistication regarding, for instance, the reliability of the solution methods and the ability to describe the intrinsic behaviour and the operating procedures of complex models. Obviously, all of these features are very important for conventional as well as non-conventional bio and chemical processes. However, the development effort that would be required to incorporate such capabilities within software packages aimed at a narrow range of applications may not be economically justifiable. Furthermore, general purpose tools are more suitable for modelling processes that incorporate a combination of conventional and non-conventional

operations.

Hence, ICAS-MoT is a mathematical modelling system that has been designed for personal computers running Microsoft Windows and that can handle a very wide range of modelling tasks in a flexible, efficient and robust manner. It is an equation-oriented modelling/simulation tool and allows the user to perform simulations of a process without having to write any programming code. External models written in text-format and/or XML-format can be imported to ICAS-MoT, which is then translated and expanded according to a RPN algorithm. The translated model can be solved, after satisfying mathematical consistency requirements, equation by equation in the debug-mode or simultaneously in the solution-mode. Statistical reports help the user in the interpretation of results. Also a sensitivity analysis option helps to analyse the effect of parameter or variables on the process behaviour. The solvable model can also be exported through a model transfer interface (COM-objects) to other simulation engines and/or external software.

All of these features of ICAS-MoT have been presented within the context of the work process related to various modelling activities during the life of a process. Some existing features have been improved and other new ones have been incorporated so that the current CAMS is more powerful and easier to use, covering a wide range of applications such as: static and dynamic simulations of either lumped or distributed process models (i.e. models represented by AE, DAE, PDE and PDAE systems), parameter estimation (static optimisation), dynamic process optimisation, and many more.

5

Application Examples

Modelling is the first step in any computer-aided study, and this chapter highlights how to formulate and propose model-based solutions through ICAS-MoT for different modelling tasks related to process/product design. ICAS-MoT has been applied for modelling tasks related to a wide range of process/product design problems. For example, in, dynamic and steady state simulation and process optimisation. The case studies also the generation of bifurcation diagrams for complex nonlinear operation of polymerisation reactors, for model parameters estimation in bio-reactors, for identifying reaction kinetics, for generating codes of physical property, for modelling separation process, for design/analysis of polymer products, for analysis of dynamic behaviour of chemical reactors and many more.

In order to illustrate the applicability of ICAS-MoT, first a few short examples are presented, followed by four case studies. These case studies deal with, model parameter identification highlight the advantages of using the developed computer-aided modelling tools.

5.1 Short Examples

In this section the use of ICAS-MoT is highlighted through a number of interesting and illustrative modelling examples, where attention is focussed on ICAS-MoT's general features in terms of functionality, tools integration and flexibility. These short examples involve thermodynamic properties calculation, dynamic simulation and control issues, dynamic model parameter identification, and process flowsheet simulation.

Most of the models used in the modelling exercises have been collected from the literature but are presented in terms of the steps of the generic modelling procedure (see section 1.4.3 in chapter one).

5.1.1 Thermodynamic Property Model

◇ **Step 1.** *System description.*
The accurate prediction of thermo-physical properties of substances is of prime importance for the design of high-performance materials and for the design and operation of efficient chemical processes. In many cases thermodynamic

models are used as stand alone models(when property values needed for a specific problem are not available though a database) or as part of a bigger model (when the property model is a subset of the total model equations as in a process simulation model). Recent attempts at developing equations of state have been based on a more rigorous statistical mechanical framework. The predictive accuracy, versatility and firm molecular foundation of the state-of-the-art SAFT (Statistical Associating Fluid Theory) approach has made it a likely candidate to replace more traditional thermodynamic models in process design. The implementation of the PC-SAFT (perturbed chain-SAFT) EOS (equation of state) is presented here to highlight how such models can be implemented, analysed, solved and (re)used through ICAS-MoT.

Figure 5.1 shows the procedure that should be followed to implement and solve the problem described above using ICAS-MoT. This problem only deals with AE (explicit and implicit).

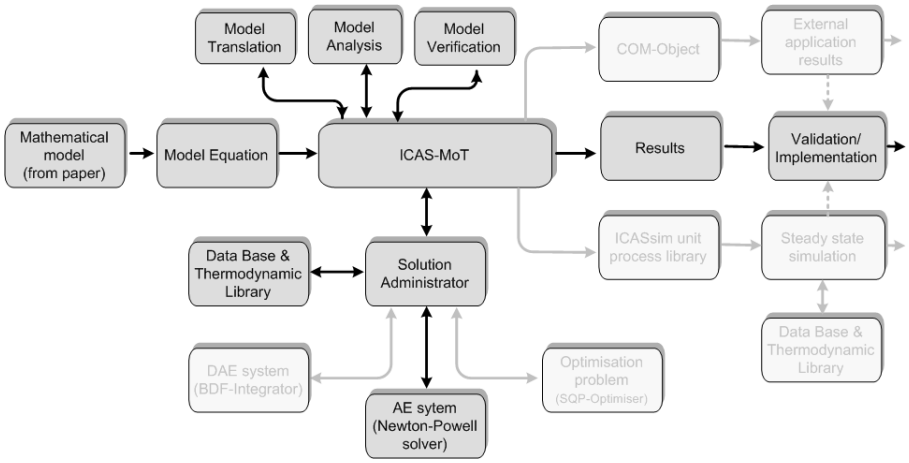


Figure 5.1: Work-Flow and tools used from ICAS-MoT to solve an AE problem

The work-flow that should be followed to solve the problem is highlighted with dark grey colour, whereas the blocks in light grey means that the path is inactivate for problems such as the one tackled here. This work-flow involves the main steps to set up the AE model and an algebraic equation solver to solve it.

◇ **Step 2.** *Problem definition.*
In the following text a summary of the model equations needed for calculating the fugacity coefficient using the PC-SAFT EOS is given. The model equations were taken from the paper by Gross and Sadowski (2001). The Helmholtz free energy A^{res} is the starting point, as all other properties can be obtained as derivatives of A^{res}. In this section, a tilde (˜) will be used for reduced quantities

and a caret symbol ($\hat{\cdot}$) will indicate molar quantities. The reduced Helmholtz free energy, for example, is given by:

$$\tilde{a}^{res} = \frac{A^{res}}{NkT}$$

At the same time, one can write it in terms of the molar quantity

$$\tilde{a}^{res} = \frac{\hat{a}^{res}}{RT}$$

Then the fugacity coefficient calculation problem is defined as, given: Temperature (T), pressure (P), mole fractions (x_i) and an initial guess of dimensionless density (packing fraction, η_{ini}). Calculate: Density (η), compressibility factor (Z_i) and fugacity coefficients (ϕ_i).

◇ **Step 3.** *Mathematical model* (PC-SAFT EOS) as derived by (Gross and Sadowski 2001).

Computation of residual Helmholtz free energy

The residual Helmholtz free energy is defined as

$$\tilde{a}^{res} = \tilde{a}^{hc} + \tilde{a}^{disp} \tag{5.1}$$

Hard-chain reference contribution is given by

$$\tilde{a}^{hc} = \bar{m}\tilde{a}^{hs} - \sum_i x_i \left(m_i - 1\right) \ln g_{ii}^{hs}\left(\sigma_{ii}\right) \tag{5.2}$$

where $\bar{m}$ is the mean segment number in the mixture

$$\bar{m} = \sum_i x_i m_i \tag{5.3}$$

The Helmholtz free energy of the hard-sphere fluid is given on a per-segment basis as:

$$\tilde{a}^{hs} = \frac{1}{\zeta_0}\left[\frac{3\zeta_1\zeta_2}{\left(1-\zeta_3\right)} + \frac{\zeta_2^3}{\zeta_3\left(1-\zeta_3\right)^2} + \left(\frac{\zeta_2^3}{\zeta_3^3} - \zeta_0\right)\ln\left(1-\zeta_3\right)\right] \tag{5.4}$$

and the radial distribution function of the hard-sphere fluid is:

$$g_{ij}^{hs} = \frac{1}{\left(1-\zeta_3\right)} + \left(\frac{d_i d_j}{d_i + d_j}\right)\frac{3\zeta_2}{\left(1-\zeta_3\right)^2} + \left(\frac{d_i d_j}{d_i + d_j}\right)^2 \frac{2\zeta_2^2}{\left(1-\zeta_3\right)^3} \tag{5.5}$$

with ζ_n defined as

$$\zeta_n = \frac{\pi}{6}\rho\sum_i x_i m_i d_i^n \qquad n = \{0,1,2,3\} \tag{5.6}$$

The temperature-dependent segment diameter d_i of component i is given by:

$$d_i = \sigma_i \left[1 - 0.12e^{-3\frac{(\varepsilon/k)_i}{T}}\right] \tag{5.7}$$

The dispersion contribution to the Helmholtz free energy is given by:

$$\tilde{a}^{disp} = -2\pi\rho I_1(\eta, \bar{m})\,\overline{m^2\varepsilon\sigma^3} - \pi\rho\bar{m}C_1 I_2(\eta, \bar{m})\,\overline{m^2\varepsilon^2\sigma^3} \tag{5.8}$$

where we have introduced an abbreviation C_1 for the compressibility expression, which is defined as

$$\begin{aligned} C_1 &= \left(1 + Z^{hc} + \rho\frac{\partial Z^{hc}}{\partial\rho}\right)^{-1} = \\ &\left\{1 + \bar{m}\frac{8\eta - 2\eta^2}{(1-\eta)^4} + (1-\bar{m})\,\frac{20\eta - 27\eta^2 + 12\eta^3 - 2\eta^4}{[(1-\eta)(2-\eta)]^2}\right\}^{-1} \end{aligned} \tag{5.9}$$

The following abbreviations also have been introduced

$$\overline{m^2\varepsilon\sigma^3} = \sum_i \sum_j x_i x_j m_i m_j \left(\frac{\varepsilon_{ij}}{kT}\right)\sigma_{ij}^3 \tag{5.10}$$

$$\overline{m^2\varepsilon^2\sigma^3} = \sum_i \sum_j x_i x_j m_i m_j \left(\frac{\varepsilon_{ij}}{kT}\right)^2\sigma_{ij}^3 \tag{5.11}$$

Conventional combining rules are employed to determine the parameters for a pair of unlike segments.

$$\sigma_{ij} = \frac{1}{2}(\sigma_i + \sigma_j) \tag{5.12}$$

$$(\varepsilon_{ij}/k) = \sqrt{(\varepsilon_i/k)(\varepsilon_j/k)}\,(1 - k_{ij}) \tag{5.13}$$

The integrals of the perturbation theory are substituted by simple power series in density

$$I_1(\eta, \bar{m}) = \sum_{i=0}^{6} a_i(\bar{m})\,\eta^i \tag{5.14}$$

$$I_2(\eta, \bar{m}) = \sum_{i=0}^{6} b_i(\bar{m})\,\eta^i \tag{5.15}$$

where the coefficients a_i and b_i depend of the chain length according to

$$a_i(\bar{m}) = a_{0i} + \frac{\bar{m}-1}{\bar{m}}a_{1i} + \frac{\bar{m}-1}{\bar{m}}\frac{\bar{m}-2}{\bar{m}}a_{2i} \quad i = 0..6 \tag{5.16}$$

$$b_i(\bar{m}) = b_{0i} + \frac{\bar{m}-1}{\bar{m}}b_{1i} + \frac{\bar{m}-1}{\bar{m}}\frac{\bar{m}-2}{\bar{m}}b_{2i} \quad i = 0..6 \tag{5.17}$$

Density

The density at a given system pressure P^{sys} must be determined iteratively by adjusting the reduced density η until $P^{calc} = P^{sys}$. A suitable starting value for a liquid phase is $\eta = 0.5$; for a vapour phase, $\eta = 10^{-10}$. Values of $\eta > 0.7405[= \pi/(3\sqrt{2})]$ are higher than the closest packing of segments and have no physical relevance.

The number density of molecules ρg is calculated from η through

$$\rho = \frac{6}{\pi}\eta \left(\sum_i x_i m_i d_i^3 \right)^{-1} \tag{5.18}$$

The quantities ζ_n given in eq. (5.6) can now be calculated. For a converged value of η, we obtain the molar density $\hat{\rho}$, in units of $[kmol/m^3]$ from:

$$\hat{\rho} = \frac{\rho}{N_{AV}} \left(10^{10} \frac{\overset{\circ}{\text{A}}}{\text{m}} \right) \left(10^{-3} \frac{\text{kmol}}{\text{mol}} \right) \tag{5.19}$$

where ρ is, according to eq.(5.18), given in units of $\overset{\circ}{\text{A}}^{-3}$ and $N_{AV} = 6.022$ x $10^{23} mol^{-1}$ denotes Avogadro's number.

PVT Relationship

Equations for the compressibility factor will be derived using the thermodynamic relation.

$$Z = 1 + \eta \left(\frac{\partial \tilde{a}^{res}}{\partial \eta} \right)_{T,x_i} = Z^{id} + Z^{hc} + Z^{disp} \tag{5.20}$$

The compressibility factor as an ideal gas contribution (id) (=1), a hard-chain contribution (hc), and a perturbation contribution, which accounts for the attractive interactions ($disp$).

The pressure can be calculated in units of $Pa = N/m^2$ by:

$$P = ZkT\rho \left(10^{10} \frac{\overset{\circ}{\text{A}}}{\text{m}} \right) \tag{5.21}$$

From eq. (5.20) and (5.1), it is

$$Z = 1 + Z^{hc} + Z^{disp} \tag{5.22}$$

The residual hard-chain reference contribution to the compressibility factor is given by:

$$Z^{hc} = \bar{m} Z^{hs} - \sum_i x_i \left(m_i - 1 \right) \left(g_{ii}^{hs} \right)^{-1} \rho \frac{\partial \ln g_{ii}^{hs}}{\partial \rho} \tag{5.23}$$

where Z^{hs} is the residual contribution of the hard-sphere fluid, given by

$$Z^{hs} = \frac{\zeta_3}{(1-\zeta_3)} + \frac{3\zeta_1\zeta_2}{\zeta_0\left(1-\zeta_3\right)^2} + \frac{3\zeta_2^3 - \zeta_3\zeta_2^3}{\zeta_0\left(1-\zeta_3\right)^3} \tag{5.24}$$

$$\rho\frac{\partial \ln g_{ij}^{hs}}{\partial\rho} = \frac{\zeta_3}{(1-\zeta_3)} + \left(\frac{d_i d_j}{d_i+d_j}\right)\left(\frac{3\zeta_2}{\left(1-\zeta_3\right)^2} + \frac{6\zeta_2\zeta_3}{\left(1-\zeta_3\right)^3}\right) + \left(\frac{d_i d_j}{d_i+d_j}\right)^2 \left(\frac{4\zeta_2^2}{\left(1-\zeta_3\right)^3} + \frac{6\zeta_2^2\zeta_3}{\left(1-\zeta_3\right)^4}\right) \tag{5.25}$$

and g_{ij}^{hs} was given in eq. (5.5)
The dispersion contribution to the compressibility factor can be written as

$$\begin{aligned} Z^{disp} &= -2\pi\rho\frac{\partial\left(\eta I_1\right)}{\partial\eta}\overline{m^2\varepsilon\sigma^3} - \\ &\pi\rho\bar{m}\left[C_1\left(\frac{\partial\left(\eta I_2\right)}{\partial\eta}\right) + C_2\eta I_2\left(\eta,\bar{m}\right)\right]\overline{m^2\varepsilon^2\sigma^3} \end{aligned} \tag{5.26}$$

where

$$\frac{\partial\left(\eta I_1\right)}{\partial\eta} = \sum_{j=0}^{6} a_j\left(\bar{m}\right)\left(j+1\right)\eta^j \tag{5.27}$$

$$\frac{\partial\left(\eta I_2\right)}{\partial\eta} = \sum_{j=0}^{6} b_j\left(\bar{m}\right)\left(j+1\right)\eta^j \tag{5.28}$$

and where C_2 is an abbreviation defined as

$$\begin{aligned} C_2 &= \left(\frac{\partial C_1}{\partial\eta}\right) \\ &= -C_1^2\left\{\bar{m}\frac{-4\eta^2+20\eta+8}{\left(1-\eta\right)^5} + \left(1-\bar{m}\right)\frac{2\eta^3+12\eta^2-48\eta+40}{\left[\left(1-\eta\right)\left(2-\eta\right)\right]^3}\right\} \end{aligned} \tag{5.29}$$

Fugacity coefficient.

The fugacity coefficient is related to the residual chemical potential according to

$$\ln\varphi_k = \frac{\mu_k^{res}\left(T,v\right)}{kT} - \ln Z \tag{5.30}$$

The chemical potential can be obtained from

$$\begin{aligned}\frac{\mu_k^{res}(T,v)}{kT} &= \tilde{a}^{res} + (Z-1) + \left(\frac{\partial \tilde{a}^{res}}{\partial x_k}\right)\Big|_{T,v,x_{i\neq k}} \\ &- \sum_j^N \left[x_j \left(\frac{\partial \tilde{a}^{res}}{\partial x_j}\right)\Big|_{T,v,x_{i\neq j}} \right]\end{aligned} \tag{5.31}$$

Where derivatives with respect to mole fractions are calculated regardless of the summation relation $\sum_j x_j = 1$. For convenience, one has defined abbreviations for derivatives of eq. (5.6) with respect to mole fraction.

$$\zeta_{n,x_k} = \left(\frac{\partial \zeta_n}{\partial x_k}\right)\Big|_{T,\rho,x_{j\neq k}} = \frac{\pi}{6}\rho m_k (d_k)^n \qquad n = \{0,1,2,3\} \tag{5.32}$$

Residual chemical potential

$$\left(\frac{\partial \tilde{a}^{res}}{\partial x_k}\right)\Big|_{T,\rho,x_{j\neq k}} = \left(\frac{\partial \tilde{a}^{hc}}{\partial x_k}\right)\Big|_{T,\rho,x_{j\neq k}} + \left(\frac{\partial \tilde{a}^{disp}}{\partial x_k}\right)\Big|_{T,\rho,x_{j\neq k}} \tag{5.33}$$

Hard-change reference contribution

$$\begin{aligned}\left(\frac{\partial \tilde{a}^{hc}}{\partial x_k}\right)\Big|_{T,\rho,x_{j\neq k}} &= m_k \tilde{a}^{hs} + \bar{m}\left(\frac{\partial \tilde{a}^{hs}}{\partial x_k}\right)\Big|_{T,\rho,x_{j\neq k}} \\ &- \sum_i x_i (m_i - 1)\left(g_{ii}^{hs}\right)^{-1}\left(\frac{\partial g_{ii}^{hs}}{\partial x_k}\right)\Big|_{T,\rho,x_{j\neq k}} - (m_k - 1)\ln g_{kk}^{hs}\end{aligned} \tag{5.34}$$

with

$$\begin{aligned}\left(\frac{\partial \tilde{a}^{hs}}{\partial x_k}\right)\Big|_{T,\rho,x_{j\neq k}} = &-\frac{\zeta_{0,x_k}}{\zeta_0}\tilde{a}^{hs} + \frac{1}{\zeta_0}\left[\frac{3\left(\zeta_{1,x_k}\zeta_2 + \zeta_1\zeta_{2,x_k}\right)}{1-\zeta_3} + \frac{3\zeta_1\zeta_2\zeta_{3,x_k}}{(1-\zeta_3)^2} + \right. \\ &\frac{3\zeta_2^2\zeta_{2,x_k}}{\zeta_3(1-\zeta_3)^2} + \frac{\zeta_2^3\zeta_{3,x_k}(3\zeta_3-1)}{\zeta_3^3(1-\zeta_3)^3} + \\ &\left(\frac{3\zeta_2^2\zeta_{2,x_k}\zeta_3 - 2\zeta_2^3\zeta_{3,x_k}}{\zeta_3^3} - \zeta_{0,x_k}\right)\ln(1-\zeta_3) + \\ &\left.\left(\zeta_0 - \frac{\zeta_2^3}{\zeta_3^2}\right)\frac{\zeta_{3,x_k}}{(1-\zeta_3)}\right]\end{aligned} \tag{5.35}$$

and

$$\left(\frac{\partial g_{ij}^{hs}}{\partial x_k}\right)\Bigg|_{T,\rho,x_{j\neq k}} = \frac{\zeta_{3,x_k}}{(1-\zeta_3)^2} + \left(\frac{d_i d_j}{d_i+d_j}\right)\left[\frac{3\zeta_{2,x_k}}{(1-\zeta_3)^2} + \frac{6\zeta_2\zeta_{3,x_k}}{(1-\zeta_3)^3}\right] + \left(\frac{d_i d_j}{d_i+d_j}\right)^2\left[\frac{4\zeta_2\zeta_{2,x_k}}{(1-\zeta_3)^3} + \frac{6\zeta_2^2\zeta_{3,x_k}}{(1-\zeta_3)^4}\right] \tag{5.36}$$

Dispersion contribution to the Helmholtz residual free energy

$$\left(\frac{\partial \tilde{a}^{disp}}{\partial x_k}\right)\Bigg|_{T,\rho,x_{j\neq k}} = -2\pi\rho\left[I_{1,x_k}\overline{m^2\varepsilon\sigma^3} + I_1\left(\overline{m^2\varepsilon\sigma^3}\right)_{x_k}\right] - \pi\rho\left\{\left[m_k C_1 I_2 + \bar{m}C_{1,x_k}I_2 + \bar{m}C_1 I_{2,x_k}\right]\overline{m^2\varepsilon^2\sigma^3} + \bar{m}C_1 I_2\left(\overline{m^2\varepsilon^2\sigma^3}\right)_{x_k}\right\} \tag{5.37}$$

with

$$\left(\overline{m^2\varepsilon\sigma^3}\right)_{x_k} = 2m_k\sum_j x_j m_j\left(\frac{\varepsilon_{kj}}{kT}\right)\sigma_{kj}^3 \tag{5.38}$$

$$\left(\overline{m^2\varepsilon^2\sigma^3}\right)_{x_k} = 2m_k\sum_j x_j m_j\left(\frac{\varepsilon_{kj}}{kT}\right)^2\sigma_{kj}^3 \tag{5.39}$$

$$C_{1,x_k} = C_2\zeta_{3,x_k} - C_1^2\left\{m_k\frac{8\eta-2\eta^2}{(1-\eta)^4} - m_k\frac{20\eta-27\eta^2+12\eta^3-2\eta^4}{\left[(1-\eta)(2-\eta)\right]^2}\right\}^{-1} \tag{5.40}$$

$$I_{1,x_k} = \sum_{i=0}^{6}\left[a_i(\bar{m})\, i\zeta_{3,x_k}\eta^{i-1} + a_{i,x_k}\eta^i\right] \tag{5.41}$$

$$I_{2,x_k} = \sum_{i=0}^{6}\left[b_i(\bar{m})\, i\zeta_{3,x_k}\eta^{i-1} + b_{i,x_k}\eta^i\right] \tag{5.42}$$

$$a_{i,x_k} = \frac{m_k}{\bar{m}^2}a_{1i} + \frac{m_k}{\bar{m}^2}\left(3-\frac{4}{\bar{m}}\right)a_{2i} \tag{5.43}$$

$$b_{i,x_k} = \frac{m_k}{\bar{m}^2}b_{1i} + \frac{m_k}{\bar{m}^2}\left(3-\frac{4}{\bar{m}}\right)b_{2i} \tag{5.44}$$

◇ **Step 4.** *Model analysis.*
A summarized set of equations and variables involved in the PC-SAFT EOS are given in Tables 5.1 and Table 5.2, respectively. Where: $n = 4$, $Ja = 7$, $Jb = 7$ and $NC =$ number of compounds.

Table 5.1: PC-SAFT model equations

Eq.	equations		Eq.	equations		Eq.	equations
5.1	1		5.16	Ja		5.32	$n * NC$
5.2	1		5.17	Jb		5.33	NC
5.3	1		5.19	1		5.34	NC
5.4	1		5.20	1		5.35	NC
5.5	NC		5.21	1		5.36	$NC * NC$
5.6	n		5.22	1		5.37	NC
5.7	NC		5.23	1		5.38	NC
5.8	1		5.24	1		5.39	NC
5.9	1		5.25	NC		5.40	NC
5.10	1		5.26	1		5.41	NC
5.11	1		5.27	1		5.42	NC
5.12	$NC * NC$		5.28	1		5.43	$Ja * NC$
5.13	$NC * NC$		5.29	1		5.44	$Jb * NC$
5.14	1		5.30	NC			
5.15	1		5.31	NC			

Table 5.2: variables of the PC-SAFT EOS model

Variable	NOV		Variable	NOV
P^{sys}	1		a_j	J_a
T	1		b_j	J_b
x_i	NC		ρ	1
m_i	NC		P	1
σ_i	NC		Z^{hc}	1
σ_i/k	NC		Z^{hs}	1
a_{oj}, a_{1j}, a_{2j}	$3 * j_a$		$\rho\left(\partial \ln g_{ii}^{hs} \middle/ \partial\rho\right)$	NC
b_{oj}, b_{1j}, b_{2j}	$3 * j_b$		Z^{dis}	1
η	1		$\partial\left(\eta I_1\right)/\partial\eta$	1
ϕ_k	NC		$\partial\left(\eta I_2\right)/\partial\eta$	1
Z	1		C_2	1
$\tilde{a}^{res}$	1		$\mu_k\left(T, v\right)/kT$	NC
$\tilde{a}^{hc}$	1		$\zeta_{n,x_k} \quad n = 0 \ldots 3$	$n * NC$
$\bar{m}$	1		$(\partial\tilde{a}^{res}/\partial x_k)_{T,\nu,x_{j\neq k}}$	NC
$\tilde{a}^{hs}$	1		$\left(\partial\tilde{a}^{hc} \middle/ \partial x_k\right)_{T,\nu,x_{j\neq k}}$	NC
g_{ij}^{hs}	NC		$\left(\partial\tilde{a}^{hs} \middle/ \partial x_k\right)_{T,\nu,x_{j\neq k}}$	NC
ζ_n	n		$\left(\partial g_{ii}^{hs} \middle/ \partial x_k\right)_{T,\nu,x_{j\neq k}}$	$NC * NC$
d_i	NC		$\left(\partial\tilde{a}^{disp} \middle/ \partial x_k\right)_{T,\nu,x_{j\neq `k}}$	NC
$\tilde{a}^{disp}$	1		$\left(m^2\varepsilon\sigma^3\right)_{x_k}$	NC
C_1	1		$\left(m^2\varepsilon^2\sigma^3\right)_{x_k}$	NC
$m^2\varepsilon\sigma^3$	1		C_{1,x_k}	NC
$m^2\varepsilon^2\sigma^3$	1		I_{1,x_k}	NC
σ_{ij}	$NC * NC$		I_{2,x_k}	NC
(ε_{ij}/kT)	$NC * NC$		a_{i,x_k}	J_a
I_1	1		b_{i,x_k}	Jb
I_2	1			

The model equations are imported to ICAS-MoT, which then translates and performs a model analysis. The results are summarized below (see Appendix D for the model code and other details).

Total number of equations:

$$NC(3NC+14)+n(NC+1)+Ja(NC+1)+Jb(NC+1)+19$$

Total number of variables:

$21[T,\ P,\eta,\ Z,\ \tilde{a}^{res},\ \tilde{a}^{hc},\bar{m},\ \tilde{a}^{hs},\ \tilde{a}^{disp},\ C_1,\ m^2\varepsilon^2\sigma^3,\ m^2\varepsilon\sigma^3,\ I_1,\ I_2,\ \rho,\ ,\ Z^{hc}$ $,\ Z^{hs}\ ,\ Z^{disp},\ (\partial(\eta I_1)/\partial\eta,\ \partial(\eta I_2)/\partial\eta\ ,\ C2]\ +\ 3NC*NC\ [\sigma_{ij},\ (\varepsilon_{ij}/kT),$ $\left(\partial g_{ii}^{hs}/\partial x_k\right)_{T,\nu,x_j\neq k}\ ]\ +\ 18\ NC\ [\ x_i,\ m_i,\ \sigma_i,\ \varepsilon_i/k,\ \phi_i,\ g_{ii}^{hs},\ d_i,\ \rho\left(\partial\ln g_{ii}^{hs}/\partial\rho\right),$ $\mu_k(T,\nu)/_{kT}\ ,(\partial\tilde{a}^{res}/\partial x_k)_{T,\nu,x_j\neq k},\ \left(\partial\tilde{a}^{hc}/\partial x_k\right)_{T,\nu,x_j\neq k},\ \left(\partial\tilde{a}^{hs}/\partial x_k\right)_{T,\nu,x_j\neq k},$ $\left(\partial\tilde{a}^{disp}/\partial x_k\right)_{T,\nu,x_j\neq k},\ \left(m^2\varepsilon\sigma^3\right)_{x_k},\ \left(m^2\varepsilon^2\sigma^3\right)_{x_k},\ C_1,\ x_k,\ I_{1,x_k},\ I_{2,x_k}\ ]$
$+\ n*(NC+1)\ [\zeta_n,\ \zeta_{n,x_k}\ ]\ +\ Ja*(NC+4)\ [a_{i,x_k}\ ,\ a_{0j},\ a_{1j},\ a_{2j},\ a_j\ ]\ +$ $Jb*(NC+4)\ [b_{i,x_k}\ ,\ b_{0j},\ b_{1j},\ b_{2j},\ b_j\ ]\ +\ NC*NC/2\ [k_{ij}\ ]$

$= NC(3NC+18)+Nz(NC+1)+Ja(NC+4)+Jb(NC+4)+NC*NC/2+21$

Degree of freedom (DOF):

The DOF is found to be: $4NC+3(Ja+Jb)+NC*NC/2+2$

The $4NC+3(Ja+Jb)+NC*NC/2+2$ variables that needs to be specified are classified as:

$(2+NC)$ variables fixed by the problem: $[T,\ P,\ x_i]$. $3NC$ fixed by the system: $[m_i,\ \sigma_i,\ \varepsilon_i/k\ ]$. $3(Ja+Jb)$ variables are fixed by the property model: $[(a_{0j},a_{1j},a_{2j}),\ (b_{0j},b_{1j},b_{2j})]$, and $NC*NC/2$ parameters regressed or retrieved from model parameter tables $[k_{ij}]$.

◇ **Step 5.** *Model data.*
For the binary system *n-Decane/Ethane* at $511.0K$ and 20.0 *bar*, find ϕ_1 and ϕ_2 when $x_1=0.3$ and $x_2=0.7$.

Input of Fixed Variables:

System: n-Decane(1)/Ethane(2)
Variables fixed by the problem: $(2+NC)=4$

$[T,P,x_1,x_2]=[511.0,20.0,0.3,0.7]$

Variables fixed by the system: $(3NC=6)$ $[m_i,\ \sigma_i,\varepsilon_i/k]$
Variables regressed or retrieved from model parameter Tables 5.3 and 5.4:

Table 5.3: Pure component parameters

Name	m	$\sigma\,[\mathring{A}]$	$\varepsilon/k\,[K]$
n-Decane	4.6627	3.8384	243.87
Ethane	1.6069	3.5206	191.42

Table 5.4: Universal model constants for equations 5.16 and 5.17

i	a_{0i}	a_{1i}	a_{2i}	b_{0i}	b_{1i}	b_{2i}
0	0.910563	-0.3084	-0.09061	0.724095	-0.57555	0.097688
1	0.636128	0.186053	0.452784	2.238279	0.69951	-0.25576
2	2.686135	-2.503	0.59627	-4.00258	3.892567	-9.15586
3	-26.5474	21.41979	-1.72418	-21.0036	-17.2155	20.64208
4	97.75921	-65.2559	-4.13021	26.85564	192.6723	-38.8044
5	-159.592	83.31868	13.77663	206.5513	-161.826	93.62677
6	91.29777	-33.7469	-33.7469	-355.602	-165.208	-29.6669

$(NC * NC/2 = 2)$ $[k12 = k21 = 0]$

variables fixed by the property model
$3(Ja + Jb) = 3(7 + 7) = 42$

◇ **Step 5.** *Model solution.* The model equations are solved in ICAS-MoT (see model code in appendix D). As output we get the following calculated variables:

$[\phi_1, \phi_2, Z] = [0.63198, 1.0160, 0.871]$

Now that the PC-SAFT EOS has been implemented and tested the code generated by ICAS-MoT, can be used by other external software or as a part of the other models within ICAS-MoT. All these have been possible by simply importing the model equation (Gross:2001) and without writing any programming code.

◇ **Step 8/9.** *Model Validation and Model transfer*

The ICAS-MoT module for fugacity coefficient using the PC-SAFT EOS that has been implemented in the previous steps is converted in a ICAS-MoT COM-Object, and used in excel for Vapour-Liquid equilibrium calculation.

The building process that follows involves the ICAS-MoT model code generation, translation, COM-Object generation, and model evaluation in external application (e.g. Excel) as is showed in the work-flow diagram given in Figure 5.2.

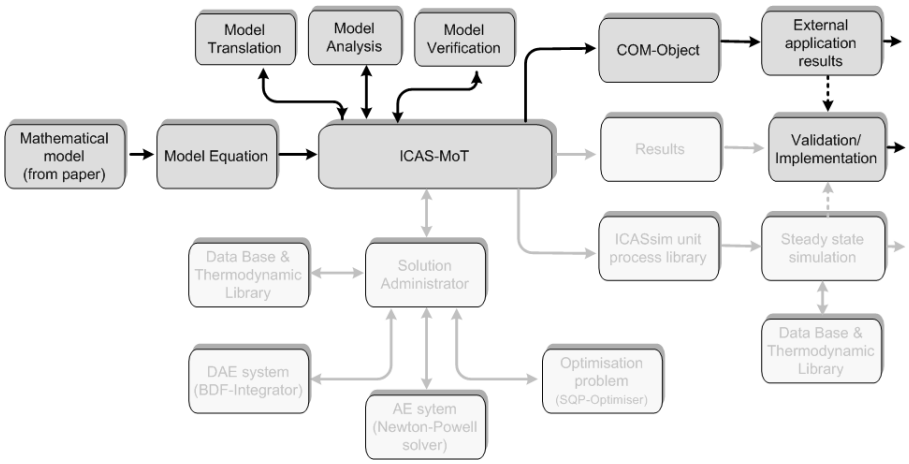


Figure 5.2: Work-Flow diagram: generating COM-Objects

The model given by Equations 5.1 - 5.44 was written as ICAS-MoT project and then was exported to EXCEL. Figure 5.3 shows the models selection from the excel macro interface.

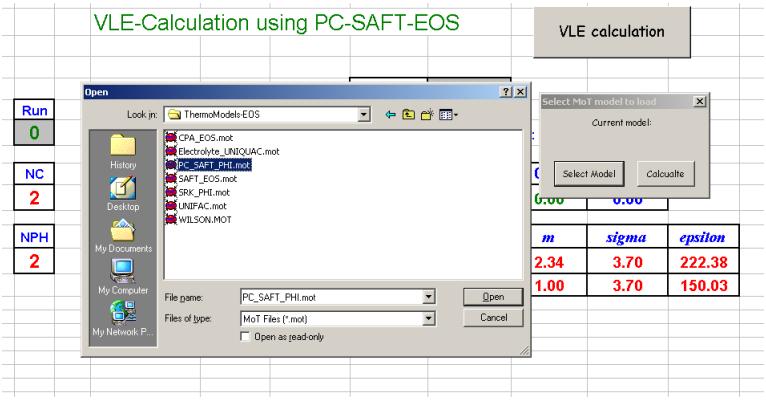


Figure 5.3: ICAS-MoT Model Selection from Excel macro interfce

The Excel macro is customised such that it is able execute a sequence of modelling related activities: reading of input data, execution of the ICAS-MoT model and writing the output (results) data, as is shown in Figure 5.2.

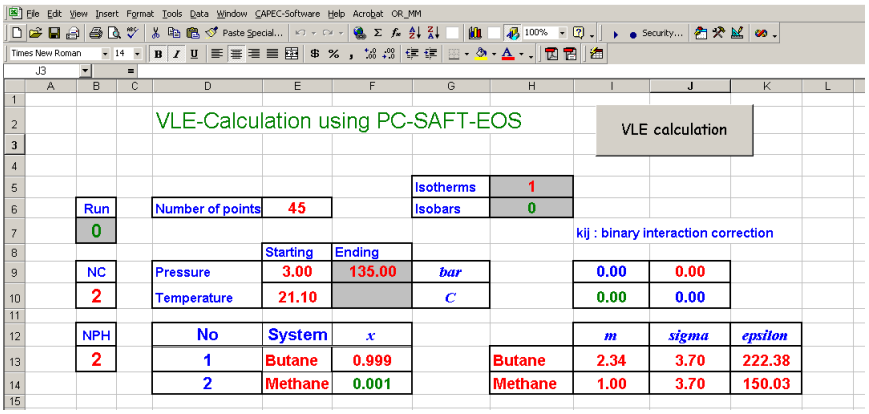


Figure 5.4: Macro-Excel interface using ICAS-MoT COM-Objects

The macro implemented in excel for VLE calculation follows the algorithm shown in figure 5.5

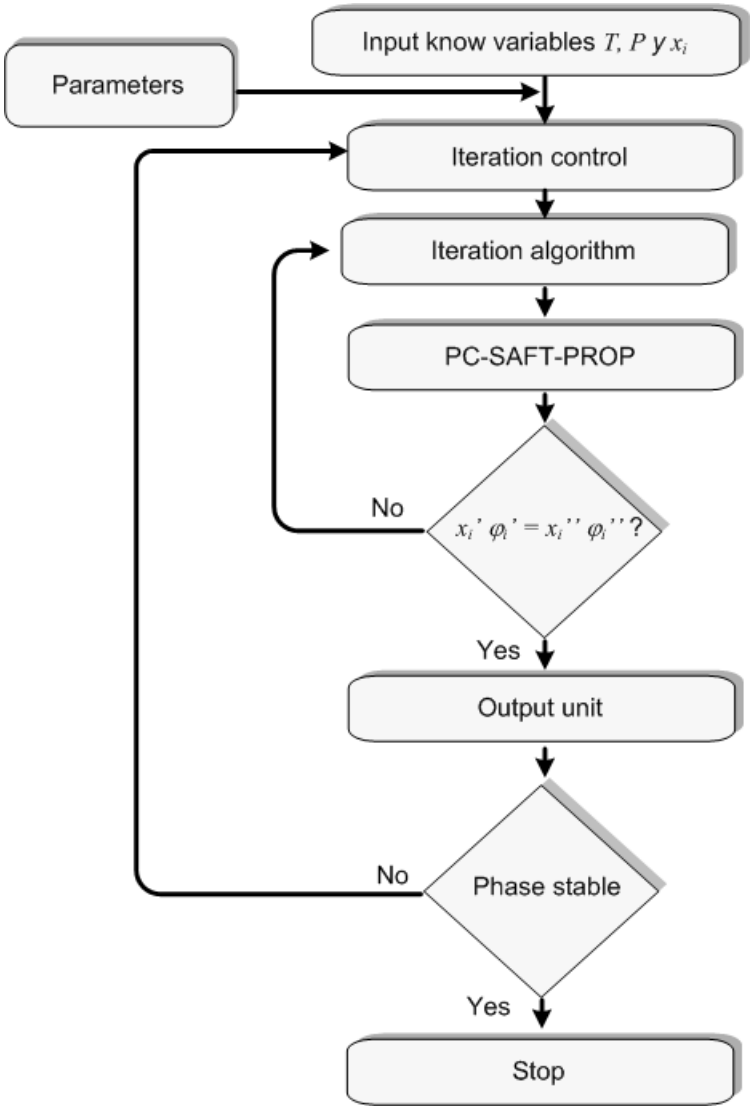


Figure 5.5: VLE calculation algorithm

Once the model is solved within Excel the macro generates the VLE data shown in the Figure 5.6

T	P	x1	x2	y1	y2
21.1	3	0.994712	0.005288	0.731457	0.268543
21.1	6	0.976094	0.023906	0.382668	0.617332
21.1	9	0.957698	0.042302	0.265495	0.734505
21.1	12	0.939509	0.060491	0.207038	0.792962
21.1	15	0.921523	0.078477	0.172196	0.827804
21.1	18	0.903734	0.096266	0.149208	0.850792
21.1	21	0.886138	0.113862	0.133021	0.866979
21.1	24	0.86873	0.13127	0.121102	0.878898
21.1	27	0.851506	0.148494	0.112047	0.887953
21.1	30	0.834461	0.165539	0.105011	0.894989
21.1	33	0.817591	0.182409	0.099459	0.900541
21.1	36	0.800891	0.199109	0.095033	0.904967
21.1	39	0.784357	0.215643	0.091488	0.908512
21.1	42	0.767984	0.232016	0.08865	0.91135
21.1	45	0.751767	0.248233	0.086392	0.913608
21.1	48	0.735701	0.264299	0.084622	0.915378
21.1	51	0.719782	0.280218	0.083268	0.916732
21.1	54	0.704004	0.295996	0.082279	0.917721
21.1	57	0.688361	0.311639	0.081614	0.918386
21.1	60	0.672849	0.327151	0.081243	0.918757
21.1	63	0.65746	0.34254	0.081144	0.918856
21.1	66	0.642188	0.357812	0.081301	0.918699
21.1	69	0.627027	0.372973	0.081703	0.918297
21.1	72	0.611969	0.388031	0.082345	0.917655
21.1	75	0.597006	0.402994	0.083224	0.916776
21.1	78	0.582128	0.417872	0.084343	0.915657
21.1	81	0.567327	0.432673	0.085706	0.914294
21.1	84	0.55259	0.44741	0.087323	0.912677
21.1	87	0.537905	0.462095	0.089206	0.910794
21.1	90	0.523259	0.476741	0.091373	0.908627
21.1	93	0.508634	0.491366	0.093845	0.906155
21.1	96	0.49401	0.50599	0.096649	0.903351
21.1	99	0.479366	0.520634	0.099818	0.900182

Figure 5.6: Calculated values in Excel

Figure 5.7 shows the calculated VLE diagram and its comparison with the experimental data.
5.10.

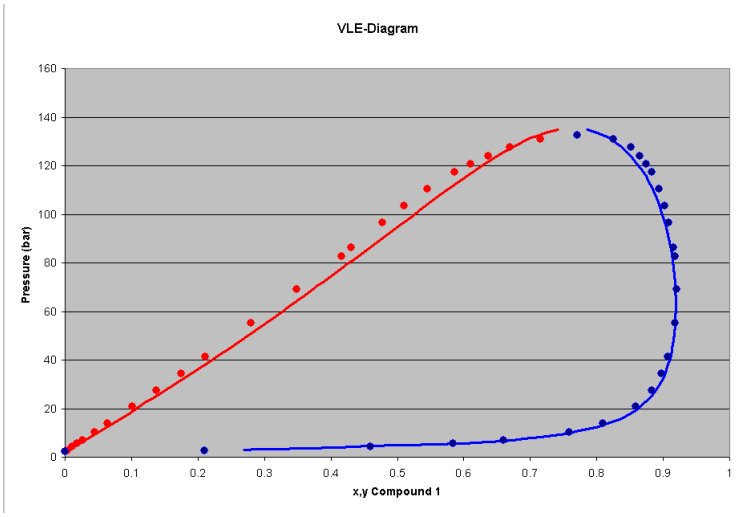


Figure 5.7: VLE-Diagriam - Plot results in Excel

In this way, the process of evaluation and use of a ICAS-MoT export model is completely automatic. The main advantage is that users can prepare data

for a ICAS-MoT export model, get output results from the ICAS-MoT export model, and store results from the same model using different parameters, directly through an Excel worksheet environment. This makes the model use easier for those not familiar with the modelling tool-box environment.

Example summary
The PC-SAFT model is a state-of-the-art, engineering-like equation of state. It is designed for modelling mixtures of all types of substances: gases, solvents, and polymers. PC-SAFT is suitable for calculation of phase equilibria and thermo-physical properties of pure components and mixtures. It has been tested against experimental data for numerous systems and found to give excellent results (Gross and Sadowski 2001). When compared to other equations of state, the PC-SAFT has be found to be more precise for correlation of experimental data and more predictive when applied to mixtures. The ICAS-MoT generated COM-Object for PC-SAFT has been used in Excel to calculate vapour liquid equilibria diagrams. Other models such as SAFT, CPA-EOS, SRK, UNIFAQ, UNIQUAC have also been implemented through ICAS-MoT.

5.1.2 Dynamic parameter estimation

◇ **Step 1.** *System description*
A better knowledge of kinetic rate constants in the modelling of chemical reactions will help in choosing operating conditions that favour the desired products. Also, heat of reaction and raw material conversion can be integrated more efficiently into process flowsheet optimisation. A set of measurements is needed to estimate parameters in the model. These measurements often contain inherent errors. For such problems maximum likelihood estimation (MLE) is often used as a method of estimation. Moreover, if the errors have a normal distribution with known covariance, MLE reduces to weighted least squares method. Under these conditions, we show that efficient optimisation techniques implemented in ICAS-MoT can be used and exploiting the least-squares structure for parameter estimation.

The work flow used to solve this problem is shown in Figure 5.8

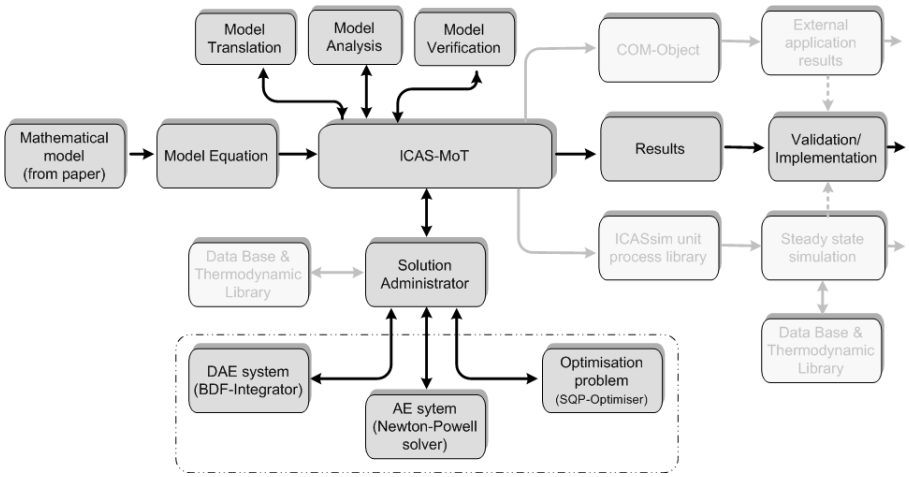
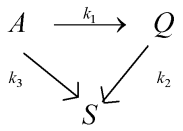


Figure 5.8: Dynamic Parameter estimation Work-Flow diagram

This problem is solved by minimizing the objective function using an optimisation method (SQP) in the outer loop, while the evaluation of the objective function and its gradients is done by numerical integration (BDF) of the ODE's in the inner loop (see Figure 3.7).

◇ **Step 2.** *Problem definition.*
The example used here describes an overall reaction of catalytic cracking of gas oil (ccgo) (A) to gasoline (Q) and other products (S). This model proposed by Froment and Bischoff (1990) is given by:

Only the concentration of A and Q were measured, therefore, the concentration of S does not appear in the model for estimation.



◇ **Step 3.** *Mathematical model.*
Modelling the measured concentrations, the differential equation model takes the form:

$$\frac{dz_1}{dt} = -\left(k_1 + k_3\right) z_1^2 \tag{5.45}$$

$$\frac{dz_2}{dt} = k_1 z_1^2 - k_2 z_2 \tag{5.46}$$

$$z_0\,[1, 0] \quad t \in [0, 0.95] \tag{5.47}$$

Where the state vector $\mathbf{z}$ is defined as $[A, Q]$, and the parameter vector θ is defined as $[k_1, k_2, k_3]$. This reaction scheme involves non-linear reaction kinetics (note that the model defined by equations 5.45 to 5.47 are non linear in the states). Given a set of experimental data the objective is to estimate θ such that the differences between the model predictions and experimental values are minimized.

◇ **Step 4.** *Model Analysis*
The model equations are imported to ICAS-MoT, which then translates and performs a model analysis. The results are summarized below (see Appendix D for the model code and other details). The model analysis given by ICAS-MOT shows that the mathematical model consist of 2 differential equation and 5 variables (3 parameters and 2 states). Then the degree of freedom is 3, which is the number of parameters that must be found by optimisation (see section 3.3.3.2).

◇ **Step 5.** *Problem data*
The data used in this example were generated using values for the parameters of $\theta = [12, 8, 2]$, with a small amount of random error (noise) added ($\sigma = 0.01$ and mean zero) and are given in table 5.5.

The initial (guess values) and bounds used for the solution of the optimisation problem are given in table 5.6

◇ **Step 6.** *Model solution*
Using the ICAS-MoT model code given in appendix D, the obtained results are shown in Table 5.7. where a comparison is done with other authors who have been also used this example to evaluate different parameter estimation

Table 5.5: Concentration data for ccgo problem

point	time	z1	z2	point	time	z1	z2
1	0.025	0.7408	0.1994	11	0.3	0.1923	0.1403
2	0.052	0.5787	0.2845	12	0.36	0.1656	0.1048
3	0.07	0.505	0.3049	13	0.4	0.1515	0.0864
4	0.1	0.4167	0.3067	14	0.46	0.1344	0.065
5	0.12	0.3731	0.2958	15	0.5	0.125	0.0541
6	0.16	0.3086	0.2616	16	0.55	0.1149	0.0433
7	0.18	0.2841	0.2423	17	0.65	0.099	0.0287
8	0.2	0.2631	0.223	18	0.7	0.0926	0.0237
9	0.22	0.2451	0.2043	19	0.75	0.087	0.0199
10	0.26	0.2155	0.1699	20	0.95	0.0667	0.0096

Table 5.6: Initial parameter values and bounds for the ccgo problem

optimisation parameter	Initial value	Lower bound	Upper bound
k_1	10	0	20
k_2	15	0	20
k_3	1	0	20

algorithms. Figure 5.9 shows the result windows from ICAS-MoT.

Table 5.7: Comparison of the optimum parameters for ccgo problem

Parameter	Real value	Esposito and Floudas, (200)	Tjoa and Biegler.(1991)	Katare et. al.(2005)	ICAS-MoT
k_1	12	12.212	11.998	12.246	11.9996
k_2	8	7.98	7.993	7.9614	7.9996
k_3	2	2.222	2.024	2.2351	2.0004
Obj. fuction	-	$2.6384\text{x}10^{-3}$	$8.221\text{x}10^{-5}$	$2.6802\text{x}10^{-3}$	$6.090\text{x}10^{-10}$

From the results shown in Table 5.7 we can say that the prediction given by the optimiser in ICAS-MoT is in accordance with the solution reported previously by other authors, and in this particular case the optimal kinetic parameter values found by ICAS-MoT are even slightly better.

◇ **Step 7/8.** *Model verification/validation*
As verification of the estimated parameter, the ICAS-MoT statistic report is given in Tables (5.8) and (5.9).

The estimated parameters are very close to the actual parameters. The measurements and the fitted values are plotted in Figure 5.10.

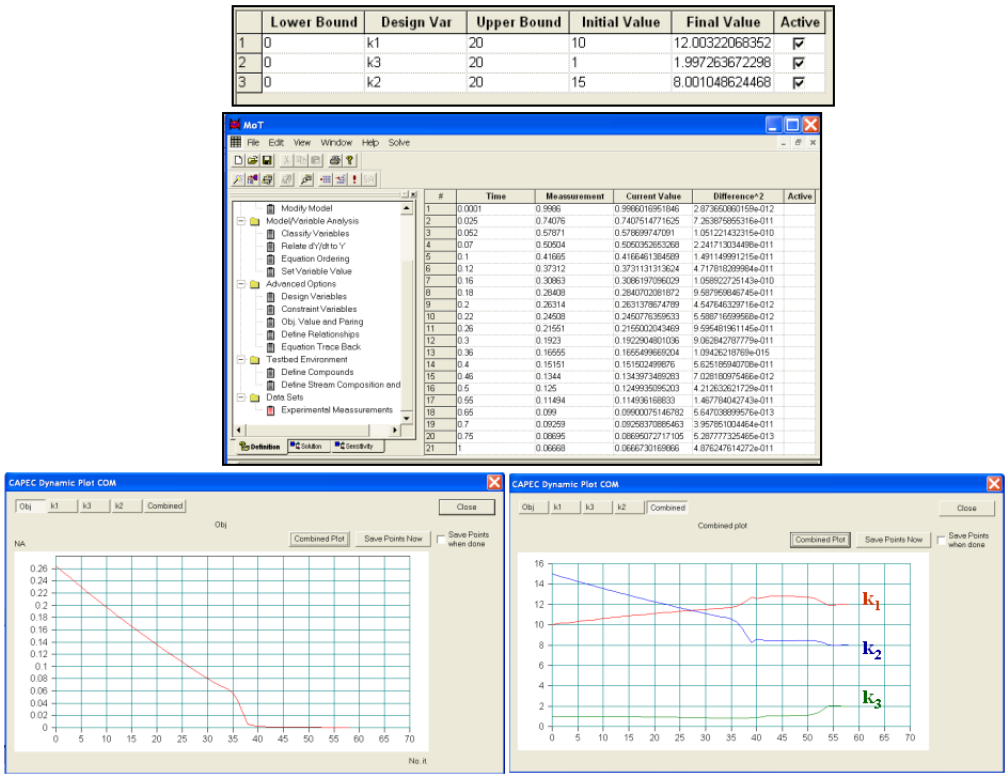


Figure 5.9: ICAS-MOT screenshot of results for the estimation kinetic paramenter of ccgo problem

Table 5.8: Statistics for the regression report generated by ICAS-MOT

Correlation:	0.999
Standard error of the estimates:	5.0×10^{-5}
R-Square:	0.999
Adjusted R-Square:	0.999
Observations:	20.000

SET 2 (z2)	
Correlation:	0.999
Standard error of the estimates:	3.8×10^{-5}
R-Square:	0.999
Adjusted R-Square:	0.999
Observations:	20.000

Table 5.9: Confidence Interval of the estimated parameters given by ICAS-MoT

Name	Value	(+/-)95% CI
k1	1.20E+01	6.21E-02
k3	2.00E+00	1.03E-02
k2	8.00E+00	4.14E-02

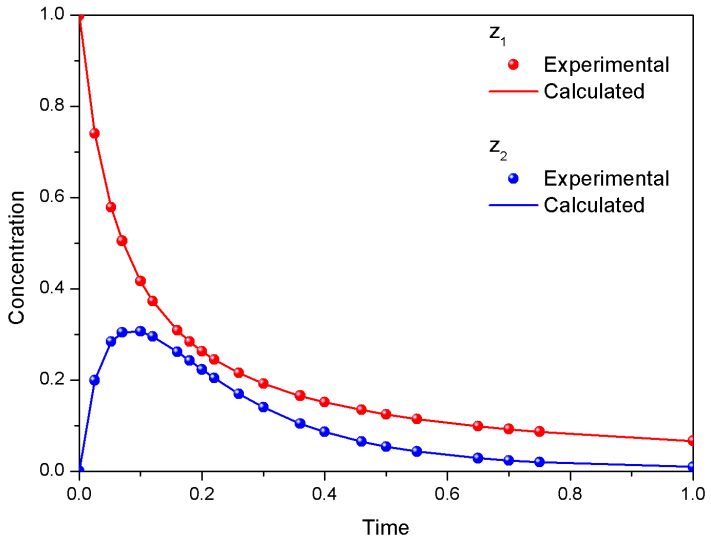


Figure 5.10: Comparison of the data and the fitted values for the catalytic cracking of gas oil problem

Example summary.
As indicated above, essentially two components are needed for the dynamic solution of parameter estimation problems: An optimisation algorithm and a differential equation solver. The subject of this example was the parameter estimation in a system modelled by ordinary differential equations. Estimating parameters in such systems is both computationally intensive as well as numerically challenging due to a variety of undesirable characteristics, such as ill-conditioning and stiffness of model equations. The goal of this example was to illustrate the systematic solution of dynamical parameter estimation problems using ICAS-MoT.

5.1.3 Simple Units and Process Models: Customised simulators

The Ethylbenzene production process has been selected as an illustrative example to highlight the generation of a customised simulator integrated within ICASSim. Ethylbenzene production is a complex process that incorporates many components and typical process equipments such as flash, drums and distillation columns. The exported models can be used as modules connected with other units/streams in a flowsheet of ICASsim.

Figure 5.11 shows the procedure that should be followed to implement and export any ICAS-MoT model to ICASsim.

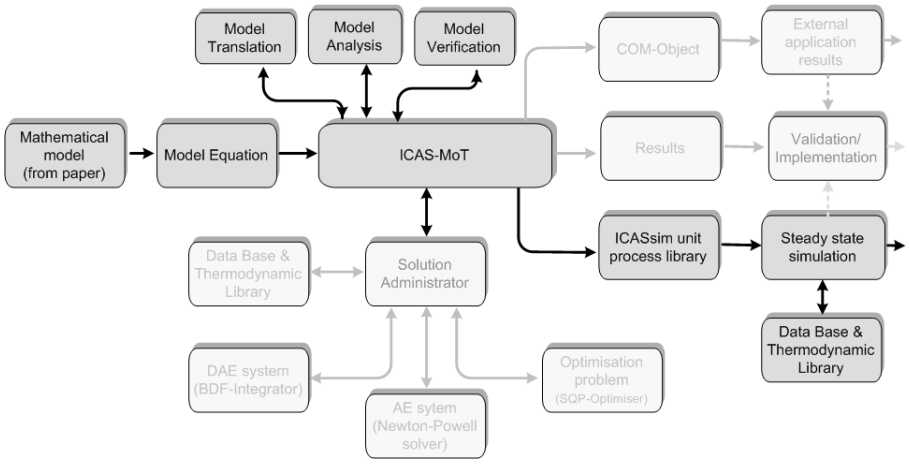


Figure 5.11: Work-flow and tools used from ICAS-MoT to export models and model aggregation

The work-flow that should be followed to implement and analyse a ICAS-MoT model and afterwards transferring it to ICASsim (process unit library) is highlighted with dark grey colour, whereas the blocks in light grey means that the path is inactivate for problems such as the one tackled here.

◇ **Step 1.** *System description*
Ethylbenzene ($C_6H_5CH_2CH_3$) is an alkylaromatic compound, almost exclusively used (99% of the production) as an intermediate for the manufacture of styrene monomer, one of the most important bulk chemicals (Cavani and Trinfir (1995))

Nowadays the Friedel-Crafts reaction (Olah (1964)) is the dominant source of Ethylbenzene from the two most commonly used routes either in the liquid or in the vapour phase. The process is carried out over an alkaline earth metal halide catalyst which is usually $AlCl_3$ an acid promoter. The reactor is operated at atmospheric pressure and the raw materials are Benzene and

Ethylene (see Figure 5.12). The alkylation reaction is exothermic and it occurs between gaseous Ethylene and liquid Benzene and its derivatives. Rates of reaction are governed by first-order kinetics in liquid-phase.

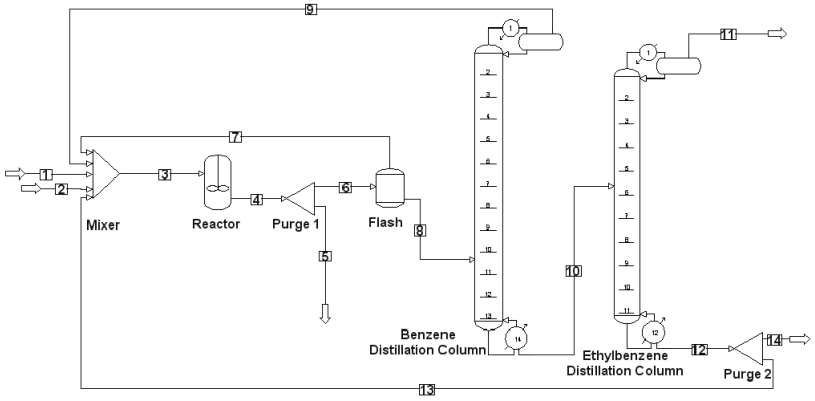


Figure 5.12: Ethylbenzene production flowsheet

Ethylene/Benzene molar ratios are adjusted to obtain an optimum yield of Ethylbenzene. As the Ethylene/Benzene ratio increases, further alkylation of Ethylbenzene occurs, leading to poly-alkylbenzenes. For all reversible secondary reactions, deliberately feeding byproduct to the reactor inhibits their formation at the source by shifting the equilibrium of the secondary reactions. This is achieved in practice by separating and recycling the byproduct, rather than separating and disposing it directly. These byproducts are polyethylbenzenes (di-Ethylbenzene, tri-Ethylbenzene, etc.) which are formed through reversible sequential reactions of Ethylbenzene.

◇ **Step 2.** *Problem definition*
The reaction scheme in the process considered is:

$$C_6H_6 + C_2H_4 \underset{k_{-1}}{\overset{k_1}{\longleftrightarrow}} C_6H_5C_2H_5$$

$$C_6H_5C_2H_5 + C_2H_4 \underset{k_{-1}}{\overset{k_1}{\longleftrightarrow}} C_6H_4\left(C_2H_5\right)_2$$

$$C_6H_4\left(C_2H_5\right)_2 + C_2H_4 \underset{k_{-1}}{\overset{k_1}{\longleftrightarrow}} C_6H_4\left(C_2H_5\right)_3$$

The reactions occur between gaseous Ethylene (E) and liquid Benzene (B) at its boiling point and its derivatives, also as liquid to obtain Ethylbenzene (EB) and higher order alkylated compounds (di-Ethylbenzene, DEB), from which only tri-Ethylbenzene (TEB) will be taken into account.

The governing reaction rates for the above reacting system are given by Smith (1996):

$$-R_B = k_1 C_B - k_{-1} C_{EB}$$

$$-R_{EB} = (k_{-1} + k_2)\, C_{EB} - (k_1 C_B + k_{-2} C_{DEB})$$

$$-R_{DEB} = (k_{-2} + k_3)\, C_{DEB} - k_{-2} C_{DEB}$$

$$-R_{TEB} = -k_{-3} C_{DEB}$$

◇ **Step 3.** *Mathematical model*
The mass and energy balances for every unit operation are described (for a steady-state basis) below:

$\mathcal{A}_1$. Steady-state conditions using a CSTR

$\mathcal{A}_2$. Complete recovery of Ethylene recycled back to the reactor

$\mathcal{A}_3$. No recycle of EB, DEB and TEB

$\mathcal{A}_4$. No purge is considered

$\mathcal{A}_5$. Equimolar feed flowrate of reactants

The above assumptions are based on the idea that, even though the recycle of the secondary products (DEB, TEB) inhibits their production, it is known that the main effect on the formation of EB will come via the principal reaction. Therefore, the most important limiting conditions, in principle, can be found through the scenario defined above (Jiménes (2005)).

► **Mixer.**
Component mass balance for the mixer:

$$x_{i,1}F_1 + y_{i,2}F_2 + y_{i,7}F_7 + x_{i,9}F_9 + x_{i,13}F_{13} - x_{i,3}F_3 = 0; i = 1, \ldots, NC \quad (5.48)$$

Note that only B and E are fed to the system. The corresponding energy balance is given by:

$$\sum_{i=1,2,7,9,13} F_j \left(\sum_{i=1}^{NC} x_{i,j} Cp_i \right) (T_j - T) = 0 \quad (5.49)$$

► **Reactor.**
Component mass balance:

$$x_{i,3}F_3 - z_{i,4}F_4 + R_k V_r = 0; i = 1, \ldots, NC \quad (5.50)$$

and the energy balance

$$F_4 \left(\sum_{i=1}^{NC} z_{i,4} Cp_i \right) (T_3 - T_4) - \sum_{k=1}^{NR} \Delta H_{r,k} R_k V_r - \dot{Q}_r = 0 \tag{5.51}$$

where

$$\Delta H_{r,k} = \sum_{i=1}^{NC} \nu_{i,k} Cp_i \left(T_4 - T_{ref} \right) - \Delta H^{\circ}_{fi,k} \tag{5.52}$$

the heat exchanged through the jacket of the reactor is modelled as

$$\dot{Q}_r = m_{cw,r} Cp_{cw} \left(T_{cw,out} - T_{cw,in} \right) \tag{5.53}$$

$$\dot{Q}_r = U A_r \left(\frac{\Delta T_1 - \Delta T_2}{\ln \left(\Delta T / \Delta T_2 \right)} \right) \tag{5.54}$$

$$\Delta T_1 = T_4 - T_{cw,in}; \Delta T_2 = T_4 - T_{cw,out} \tag{5.55}$$

► **Purge 1 process model.**
Total mass balance:

$$F_4 = F_5 + F_6 \tag{5.56}$$

with purge fraction defined as $\sigma_1 = F_6/F_4$. Note that the concentrations and temperatures on streams 4, 5 and 6 are the same, that is:

$$z_{i,4} = z_{i,5} = z_{i,6}; i = 1, \ldots, NC \tag{5.57}$$

$$T_4 = T_5 = T_6 \tag{5.58}$$

► **Component (Phase) splitter.**
This unit will be modelled as an isothermal flash drum. The overall mass balance is:

$$F_6 = F_7^v + F_8^l \tag{5.59}$$

and the vapour-liquid equilibrium is described simply by:

$$K_i = \frac{p_{i,F}^{sat}}{P_F}, \qquad p_{i,F}^{sat} = \exp \left(A_i + \frac{B_i}{C_i + T_6 - T_{ref}} \right) \tag{5.60}$$

$$y_{i,7} = K_i x_{i,8}; \qquad i = 1, \ldots, NC \tag{5.61}$$

$$x_{i,8} = \frac{z_{i,6}}{1 - \beta_F \left(1 - K_i \right)} \tag{5.62}$$

where

$$\beta_F = F_7^v / F_6 \tag{5.63}$$

where pressure is given in mmHg.

► **Benzene Column.**
For simplicity, the separation task in the distillation columns will be characterized by simply setting the amounts of light and heavy key components (LK_c and HK_c, respectively) to be recovered at the column. Therefore, flowrates, compositions and temperatures at the top and bottom can be obtained. For this column, Benzene is selected as the light component and Ethylbenzene as the heavy key component. Thus, overall mass balance:

$$F_8 = F_9 + F_{10} \tag{5.64}$$

Balance per component:

$$x_{i,8} F_8 = y_{i,9} F_9 + x_{i,10} F_{10}; \qquad i = 1, \ldots, NC \tag{5.65}$$

Dew Point (top of the column):

$$1 = \sum_{i=E}^{B} x_{i,9} = \sum_{i=E}^{B} y_{i,9} P_{BC} / p_{i,BC}^{sat}\,(T_9) \tag{5.66}$$

Bubble point (bottom of the column):

$$1 = \sum_{i=B}^{NC} y_{i,10} = \sum_{i=B}^{NC} x_{i,10} p_{i,BC}^{sat}\,(T_{10}) / P_{BC} \tag{5.67}$$

► **Ethylbenzene Column.**
In this column the Ethylbenzene is assumed to be the light-key component (LK_{EB}) while the di-Ethylbenzene is considered to be the heavy-key (HK_{EB}) component. Hence, overall mass balance:

$$F_{10} = F_{11} + F_{12} \tag{5.68}$$

Balance per component:

$$x_{i,10} F_{10} = y_{i,11} F_{11} + x_{i,12} F_{12}; \qquad i = 1, \ldots, NC \tag{5.69}$$

Dew Point (top of the column):

$$1 = \sum_{i=E}^{EB} x_{i,11} = \sum_{i=E}^{EB} y_{i,11} P_{EBC} / p_{i,EBC}^{sat}\,(T_{11}) \tag{5.70}$$

Bubble point (bottom of the column):

$$1 = \sum_{i=EB}^{NC} y_{i,12} = \sum_{i=EB}^{NC} x_{i,12} p_{i,EBC}^{sat}\left(T_{12}\right)/P_{EBC} \tag{5.71}$$

▶ **Purge 2 process model.**
Total mass balance:

$$F_{12} = F_{13} + F_{14} \tag{5.72}$$

with purge fraction defined as $\sigma_2 = F_{13}/F_{12}$. Note that the concentrations and temperatures on streams 12, 13 and 14 are the same, that is:

$$x_{i,12}{=}x_{i,13}{=}x_{i,14}; \qquad i = 1, \ldots, NC \tag{5.73}$$

$$T_{12}{=}T_{13}{=}T_{14} \tag{5.74}$$

◇ **Step 4. Model analysis.**

The process model can be presented in the following generic form:

$$\mathbf{0} = \mathbf{g}\left[\mathbf{x}, \mathbf{u}, \mathbf{p}\right] \tag{5.75}$$

$$\mathbf{x_s} = \left[\begin{array}{l} F_3, F_4, F_8, T_3, T_4, T_{cw,out}, T_9, T_{10}, T_{11}, T_{12}, \\ \beta_F, z_{i,3}, z_{i,4}, x_{i,8}, y_{i,9}, x_{i,10}, y_{i,11}, x_{i,12} \\ \qquad i = 1, \ldots, NC \end{array}\right]^T \tag{5.76}$$

$$\mathbf{x_p} = \left[\begin{array}{l} F_5, F_6, F_7, F_9, F_{10}, F_{11}, F_{12}, F_{13}, F_{14}, T_5, T_6, T_{cw,out}, \\ \dot{Q}_r, x_B, x_E, p_{EB}, y_E, \beta_{Y,S}, \alpha_{Y,S}, \delta_{Y,S}, \tau_{Y,S}, \varepsilon_{Y,S}, \\ R_k, z_{i,5}, z_{i,6}, y_{i,7}, x_{i,9}, y_{i,10}, x_{i,11}, y_{i,12}, x_{i,13}, x_{i,14}, \\ K_i, p_{i,F}^{sat}, p_{i,BC}^{sat}, p_{i,EBC}^{sat}, \Delta H_{r,k} \\ \qquad i = 1, \ldots, NC, k = 1, \ldots, NR \end{array}\right]^T \tag{5.77}$$

$$\mathbf{u} = \left[F_1, f_E, m_{cw,r}, LK_c, HK_c, x_{i,1}, y_{i,2}\right]^T; c = BC, EBC \tag{5.78}$$

$$\mathbf{p} = \left[\begin{array}{l} UA_r, V_r, Cp_{cw}, P_F, P_{BC}, P_{EBC}, T_1, T_2, T_{cw,in}, T_{ref}, \\ \sigma_1{,}\sigma_2{,}A_i, B_i, C_i, \Delta H_{f,i}^{\circ}, Cp_i, k_k \\ \qquad i = 1, \ldots, NC, k = 1, \ldots, NR \end{array}\right]^T \tag{5.79}$$

where vector $\mathbf{x}$ contains the 46 states ($\mathbf{x_s}$) and 81 process variables ($\mathbf{x_p}$). The 9 known inputs (design variables) are contained in vector $\mathbf{u}$, and vector $\mathbf{p}$ contains 42 parameters. The system consists of 204 variables and 153 equations between state and algebraic equations. Therefore, DOF = 51, meaning that all 9 design variables $\mathbf{u}$ plus the 42 parameters $\mathbf{p}$ must be given.

◇ **Step 5.** *Model data.*

Table 5.10 lists the physical properties of the components, while Table 5.11 shows the process specifications utilised for the simulations (Jiménes (2005)).

Table 5.10: Components physical properties.

Component	Heat Capacity (kJ/kmol/K)	Heat of formation (kJ/kmol)
B	137.000	82966.520
EB	182.312	29804.320
DEB	239.061	−21855.001
TEB	310.687	−70613.998
E	43.170	52308.256

Vapour Pressure (Antoine equation):
$P = \exp\left[A_i + B_i/(C_i + T(^\circ C) + T_{ref})\right]$

Component	Constant A	Constant B	Constant C
B	15.84	−2755.64	219.16
EB	16.04	−3291.66	213.77
DEB	16.80	−4170.18	226.41
TEB	16.39	−4214.88	213.92
E	15.80	−1420.40	258.69

Table 5.11: Process specifications for the *design* target.

Inlet Conditions			**Reactor Conditions**		
$x_{B,1}F_1$	=	10.08 kmol/h	V_r	=	4.74 m^3
$y_{E,2}F_2$	=	10.27 kmol/h	$m_{cw,r}$	=	2500 kmol/h
$T_1 = T_2$	=	353.15 K	$T_{cw,in}$	=	293.15K
σ_1	=	0.90	UA_r	=	4.594×10^5kJ/K/h
σ_2	=	0.91	$k_1{=}k_{-1}{=}k_2{=}k_{-2} = 0.4$ h^{-1}, k_3=0.02 h^{-1}		
Benzene Column			**Ethylbenzene Column**		
LK_{BC}	=	0.9926	LK_{EBC}	=	0.99994
HK_{BC}	=	2.5303×10^{-2}	HK_{EBC}	=	1.00×10^{-3}
P_{BC}	=	760 mmHg	P_{EBC}	=	760 mmHg

◇ **Step 6.** *Model Transfer and external simulation.*

In this example the main idea is to generate a model of each unit as a module, and then connect them in an ICAS flowsheet for its simulation. First the corresponding ICAS-MoT models for this process are created (see model codes

on the Appendix D). Then the following steps must be performed in order to make the flowsheet simulation in the ICAS environment:

1. Export each ICAS-MoT model to ICASsim unit library, using the export option available in ICAS-MoT.

2. Draw the flowsheet choosing the properly units (adding ICASsim Units, ICAS-MoT units, and input and output streams). The unit must have the same number of input and output streams as assigned to the model at the time when it was exported from ICAS-MoT. Figure 5.13 shows the entire flowsheet using the ICAS-MoT models units for the Ethylbenzene production process.

3. Specify components, thermodynamic models, and define stream composition. For each input stream, pressure, temperature and flows must be given.

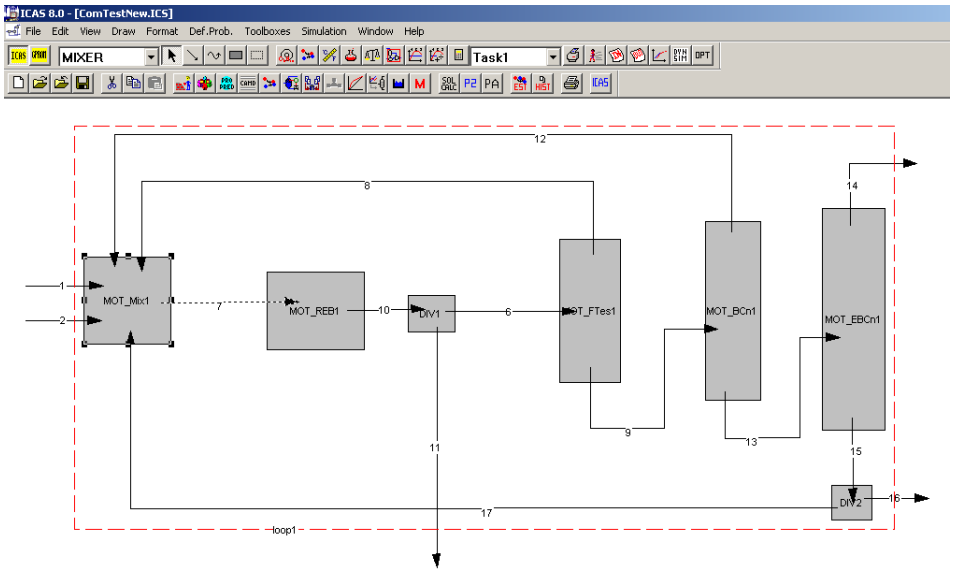


Figure 5.13: ICAS-MoT models imported and used in ICASsim for the Ethylbenzene production process.

◇ **Step 7.** *Model solution.*

Once the above steps have been finished the flowsheet is ready to be solved. The "Perform the simulation" button in the ICAS environment lead to generate the model solution. Figure 5.14 shows the simulation results.

STREAM NUMBER	1	2	3	4
TEMPERATURE (K)	353.15000	353.15000	338.75561	300.00000
PRESSURE (atm)	1.00000	1.00000	1.00000	1.00000
ENTHALPY(K/kmole)	6834.89308	6613.53359	7388.63980	4455.00753
ENTROPY(1/kmole)	23.69749	27.28365	30.52416	26.14180
U-ENERGY(K/kmole)	6834.78814	6584.64349	7365.51479	4444.92805
DENS. (kmole/m^3)	9.52966	0.03461	0.04324	0.09921
VAPOUR FRACTION	0.00000	1.00000	0.84249	0.40989
LIQUID FRACTION	1.00000	0.00000	0.15751	0.59011
--------(kmole/hr)----------				
ETHYLENE	0.00000	10.27000	27.42549	20.11195
Benzene	10.08000	0.00000	30.42760	24.00982
ETHYLBENZENE	0.00000	0.00000	0.15304	5.73389
m-diethylbenzene	0.00000	0.00000	2.57246	3.35056
1,2,4-Triethylbe	0.00000	0.00000	0.20408	0.26291
TOTAL	10.08000	10.27000	60.78268	53.46913

STREAM NUMBER	5	6	7	8
TEMPERATURE (K)	300.00000	300.00000	300.00000	300.00000
PRESSURE (atm)	1.00000	1.00000	1.00000	1.00000
ENTHALPY(K/kmole)	4455.00753	4455.00753	6668.02672	2917.86181
ENTROPY(1/kmole)	26.14180	26.14180	27.36756	25.29040
U-ENERGY(K/kmole)	4444.92805	4444.92805	6643.60367	2917.74526
DENS. (kmole/m^3)	0.09921	0.09921	0.04094	8.57958
VAPOUR FRACTION	0.40989	0.40989	1.00000	0.00000
LIQUID FRACTION	0.59011	0.59011	0.00000	1.00000
--------(kmole/hr)----------				
ETHYLENE	2.95646	17.15549	16.80213	0.35337
Benzene	3.52944	20.48038	1.83783	18.64254
ETHYLBENZENE	0.84288	4.89101	0.05010	4.84091
m-diethylbenzene	0.49253	2.85803	0.00453	2.85350
1,2,4-Triethylbe	0.03865	0.22426	0.00004	0.22422
TOTAL	7.85996	45.60917	18.69463	26.91454

STREAM NUMBER	9	10	11	12
TEMPERATURE (K)	352.97919	419.50000	409.15492	454.15751
PRESSURE (atm)	1.00000	1.00000	1.00000	1.00000
ENTHALPY(K/kmole)	10463.55594	-828.98367	5708.50304	-3471.28666
ENTROPY(1/kmole)	34.17522	42.56725	48.90030	49.39460
U-ENERGY(K/kmole)	10435.39380	-829.16317	5676.09313	-3471.50481
DENS. (kmole/m^3)	0.03551	5.57114	0.03085	4.58409
VAPOUR FRACTION	1.00000	0.00000	1.00000	0.00000
LIQUID FRACTION	0.00000	1.00000	0.00000	1.00000
--------(kmole/hr)----------				
ETHYLENE	0.35337	0.00000	0.00000	0.00000
Benzene	18.50860	0.13394	0.13394	0.00000
ETHYLBENZENE	0.02020	4.82071	4.72300	0.09772
m-diethylbenzene	0.00000	2.85350	0.02674	2.82675
1,2,4-Triethylbe	0.00000	0.22422	0.00000	0.22422
TOTAL	18.88217	8.03237	4.88368	3.14869

STREAM NUMBER	13	14
TEMPERATURE (K)	453.79976	453.79976
PRESSURE (atm)	1.00000	1.00000
ENTHALPY(K/kmole)	-3464.96424	-3464.96424
ENTROPY(1/kmole)	49.34366	49.34366
U-ENERGY(K/kmole)	-3465.18212	-3465.18212
DENS. (kmole/m^3)	4.58959	4.58959
VAPOUR FRACTION	0.00000	0.00000
LIQUID FRACTION	1.00000	1.00000
--------(kmole/hr)----------		
ETHYLENE	0.00000	0.00000
Benzene	0.00000	0.00000
ETHYLBENZENE	0.09857	0.00975
m-diethylbenzene	2.56793	0.25397
1,2,4-Triethylbe	0.20404	0.02018
TOTAL	2.87054	0.28390

THE TIME SPENT IN IS: 0.92200 s

TIME : 09:54:47

Figure 5.14: Simulation results from ethyl-benzene production –stream report

Example summary
The process model considered in this example does not incorporate rigorous unit operation models because the main purpose was to show how to generate a customised simulator by creating individual modules (one for each unit) and then using them in an ICAS flowshet. However, the models are accurate enough to characterise the overall process performance. Here all the unit modules were generated in ICAS-MoT, but also some units already defined in ICAS environment can be interconnected, leading to the (either steady-state or dynamic) simulation of a wide range of chemical processes.

5.2 Case Studies

A number of bio and chemical process models have been implemented, tested and then used for further study of the process/product through ICAS-MoT. Here four representative case studies have been selected for detailed presentation. These include: (a) Kinetic parameter identification in an experimental bio-reaction system, (b) steady state analysis, dynamic simulation and process operation optimisation of a co-polymerisation reactor, (c) modelling and design of molecular distillation processes (distributed systems), and (d) the simulation of the well-known Tennessee Eastman process (a plant-wide model). In the following sections these case studies are treated at length.

5.2.1 Model Identification: Anaerobic Bio-gas process

Anaerobic digestion is a widely used method for the treatment of sewage sludge. Several steps have been taken towards establishing general purpose and comprehensive mechanistic models of bioprocesses, involving a large number of reactions with highly non-linear kinetics. However, as many of the kinetic parameters are unknown, they must be identified and calibrated before the model can be used for the enhancement of the organic matter removal. The present case study deals with two-step (a short hydrolytic step followed by a second longer methanogenic step) anaerobic digestion of primary and secondary sludge in order to establish the actual kinetic mechanism of the different microbial processes and through this, the development of a reliable (dynamic) process model, which can be used for simulation and optimisation studies related to the production of the biogas and effluent quality. In particular, the modelling aspects (i.e. model construction, model analysis, model parameter identification and dynamic simulation) are integrated through the use of computer-aided modelling system ICAS-MoT.

Figure 5.15 shows the tools that should be used to implement and solve the problem described above using ICAS-MoT. This problem deals with dynamic parameter estimations [DAEs (AEs + ODEs) and optimisation procedure).

The work-flow that should be followed to solve the problem is highlighted with dark grey colour, whereas the blocks in light grey means that the path is inactivate for problems such as the one tackled here. This work-flow involves the main steps to set up the DAE-Optimisation model solvers to solve it.

As a result, a trustworthy kinetic characterization validated against several experimental data is obtained. Furthermore, the results highlight the advantages of using a computer-aided modelling system in terms of time and modelling effort. In particular the model analysis that assures the model confidence, the parameter sensitivity and the model statistics using experimental data is performed. This step corresponds to a diagnostic checking and may involve statistical analysis of the fitted (optimised) model parameters, which can help the experimental design with minimal effort(Asprey and Macchietto (2002)).

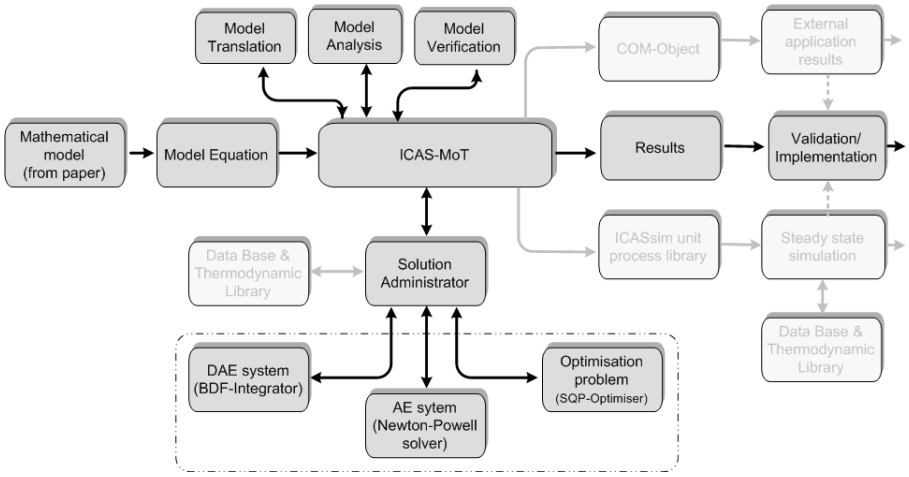


Figure 5.15: Work-flow and tools used from ICAS-MoT for dynamic parameter identification (case study 1)

◇ **Step 1.** *System description*
Anaerobic digestion is an appropriate technique for the treatment of sludge before final disposal and it is employed worldwide as the oldest and most important process for sludge stabilization. During this process insoluble organic matter is hydrolyzed and subsequently fermented to short-chain volatile fatty acids (VFAs) (e.g. acetate, propionate, butyrate, iso-butyrate, etc.), H_2 and CO_2 by fermentative bacteria. VFAs are oxidized by syntrophic bacteria to acetate, H_2 and CO_2, and finally the latter compounds are converted by methanogenic archaea to biogas (methane plus carbon dioxide). Compared with other treatment methods (i.e. landfill disposal, incineration, composting), anaerobic digestion does not have only the advantage of energy (methane) production but furthermore is characterized by sludge disinfection that makes possible the reuse of the effluent as fertilizer especially when the process is carried out at thermophilic temperatures. Thus, anaerobic digestion is a sustainable method for the treatment of sewage sludge and well worth studying further. Usually, the hydrolysis is the rate-limiting step during the anaerobic digestion of primary and/or secondary sludge. It has been proved that the thermal pretreatment of sludge at elevated temperature (100 up to 275 C) significantly increases the disintegration and solubilization of sludge solids and thus improves the sludge stabilization. However, high temperature pre-treatment has high-energy requirements and is difficult to be operated. Therefore, two-step processes (with a first short hydrolytic step at temperatures below 100 C and a second longer methanogenic step at temperatures between 35 and 55 C) become more attractive. To date, there are several studies showing the effectiveness of a lower temperature (60-100 C) in the first step of the two-step anaerobic di-

gestion of sludge. Most of these studies focus on the investigation of the choose of temperature and the duration of the first step. In general, mesophilic anaerobic digestion of sewage sludge is more widely used compared to thermophilic digestion, mainly because of the lower energy requirements and higher stability of the process. However, the thermophilic anaerobic digestion process is characterized by increased methanogenic potential at lower hydraulic retention times. Recently, the effect of a 70 C first step on the outcome of mesophilic and thermophilic anaerobic digestion of primary and secondary sludge has been examined. Therefore the main scope of the present case study is to investigate the two step (first short hydrolytic step at 70 C and a second longer methanogenic step at 55 C) anaerobic digestion of primary and secondary sludge in order to extract the kinetic characteristics of the different microbial processes, so that afterwards the sludge digestion process can be adequately simulated. Thus, the aim of this case study is the modelling, kinetic parameter identification and simulation of an experimental two-step (a first short hydrolytic step at 70 C followed by a second longer methanogenic step at 55 C) anaerobic digestion of primary and secondary sludge, using ICAS-MoT. For this purpose first kinetic experiments using primary and secondary sludge were carried out according to a predefined set of scenarios; then an initial kinetic mechanism is proposed; afterwards a strategy of solution is proposed by dividing the parameter identification problem into several sub-problems, where the available data is used to estimate the unknown kinetic model parameter; and finally the kinetic characterization is validated achieving a corrected kinetic mechanism for the two step anaerobic digestion process.

◇ **Step 2.***Problem definition*

Kinetic mechanism. The initial kinetic mechanism proposed in this work is schematically shown in the Figure 5.16. Previous kinetic studies have shown that the anaerobic decomposition of organic matter to methane (CH_4) and carbon dioxide (CO_2) passes through the production of volatile fatty acids (VFAs) such as butyrate (But) and propionate (Pr), acetate (Ac), hydrogen (H_2) and "other" soluble organic components. According to Figure 5.16, the methane production from the particulate organic matter consists of the following steps:

(a) hydrolysis of the organic particles to acetic acid and "other" soluble organic components,

(b) fermentation of "others" to butyric, propionic and acetic acid,

(c) fermentation of butyric and propionic acid to acetic acid, and

(d) conversion of acetic acid and hydrogen (produced in the above steps) to methane.

Based on the mechanism depicted in Figure 5.16, the series-parallel reactions are given by:

$$solids \quad \rightarrow \quad Others + \ Ac \ + \ H_2 \tag{5.80}$$

$$Others \quad \rightarrow \quad VFAs \ + \ Ac \ + \ H_2 \tag{5.81}$$

$$\mathrm{Pr} \quad \rightarrow \quad Ac \ + \ H_2 \tag{5.82}$$

$$But \quad \rightarrow \quad Ac \ + \ H_2 \tag{5.83}$$

$$Ac \quad \rightarrow \quad CH_4 \ + \ CO_2 \tag{5.84}$$

$$4H_2 \ + \ CO_2 \quad \rightarrow \quad CH_4 \ + \ H_2O \tag{5.85}$$

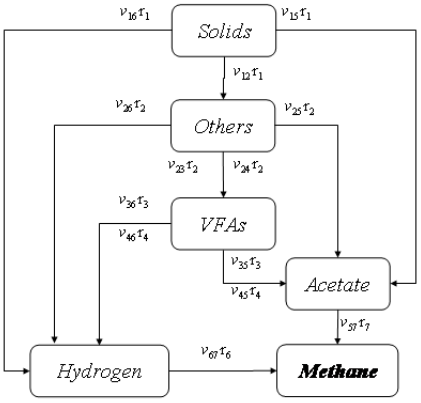


Figure 5.16: Kinetic mechanism for anaerobic bioconversion of organic matter to methane.

◇ **Step 3.** *Model construction*

We are interested in developing a model for the biodegradation system in the two CSTRs in series. In the bioprocess, the controlling mechanism includes reaction kinetics, fluid flow and retention time for the batch case. The following initial assumptions have been made:

(a) the system operates at constant temperature (isothermal),

(b) the volume reactor is constant,

(c) the reactor system is well-mixed, and

(d) there is no physical loss in the system.

Conservation Equations
Taking into account the assumptions mentioned before, the mass balances for each component becomes:

(a) In the liquid phase

$$\frac{d[Solids]}{dt} = F \cdot \Big([Solids]_f - [Solids]\Big) - r_1 \qquad [Solids]_{t=0} = [Solids]_0 \tag{5.86}$$

$$\frac{d[Others]}{dt} = F \cdot \Big([Others]_f - [Others]\Big) + (-r_2 + \nu_{12} \cdot r_1) \qquad [Others]_{t=0} = [Others]_0 \tag{5.87}$$

$$\frac{d[\mathrm{Pr}]}{dt} = F \cdot \Big([\mathrm{Pr}]_f - [\mathrm{Pr}]\Big) + (-r_3 + \nu_{23} \cdot r_2) \qquad [\mathrm{Pr}]_{t=0} = [\mathrm{Pr}]_0 \tag{5.88}$$

$$\frac{d[But]}{dt} = F \cdot \Big([But]_f - [But]\Big) + (-r_4 + \nu_{24} \cdot r_2) \qquad [But]_{t=0} = [But]_0 \tag{5.89}$$

$$\frac{d[Ac]}{dt} = F \cdot \Big([Ac]_f - [Ac]\Big) + (-r_5 + \nu_{45} \cdot r_4 + \nu_{35} \cdot r_3 + \nu_{25} \cdot r_2 + \nu_{15} \cdot r_1) \qquad [Ac]_{t=0} = [Ac]_0 \tag{5.90}$$

(b) In the gas phase

$$\frac{dP_{H2}}{dt} = \left(\frac{V_L \cdot R_g \cdot T}{V_G \cdot 1000} \cdot \gamma\right) \cdot (-\nu_{66} \cdot r_6 + \nu_{46} \cdot r_4 + \nu_{36} \cdot r_3 + \nu_{26} \cdot r_2 + \nu_{16} \cdot r_1) \tag{5.91}$$

(c) Total methane production

$$\frac{d\,[CH_4]}{dt} = V_L \cdot \delta \cdot (v_{57} \cdot r_5 + v_{67} \cdot r_6), \qquad [CH_4]_{t=0} = [CH_4]_0 \tag{5.92}$$

Constitutive Equations According to Eqs. 5.80 - 5.85, the reaction rate expressions are given by:

$$r_1 = k_h\,(T) \cdot [Solids] \tag{5.93}$$

$$r_3 = \frac{u_{\max 3} \cdot [Pr]}{k_{s3} + [Pr]} \cdot X_T \tag{5.94}$$

$$r_3 = \frac{u_{\max 3} \cdot [Pr]}{k_{s3} + [Pr]} \cdot X_T \tag{5.95}$$

$$r_4 = \frac{u_{\max 4} \cdot [But]}{k_{s4} + [But]} \cdot X_T \tag{5.96}$$

$$r_5 = \frac{u_{\max 5} \cdot [Ac]}{k_{s5} + [Ac]} \cdot X_T \tag{5.97}$$

$$r_6 = \frac{u_{\max 6} \cdot p_{H_2}}{k_{s6} + p_{H_2}} \cdot X_T \tag{5.98}$$

$$p_{H_2} = [H_2] \cdot R_g \cdot T \tag{5.99}$$

◇ **Step 4.** *Model analysis*

The model Analysis procedure is performed using ICAS-MoT. As a first step, the model equations are imported to ICAS-MoT, which then translates and performs a model analysis. According to the process model given by equations 5.86 - 5.99, the total number of variables (Nv) is 49, which can be classified as follows: (a) 7 states: $[Solids]$, $[Others]$, $[Pr]$, $[But]$, $[Ac]$, $[H_2]$, Q_{CH4}; (b) 16 auxiliary variables: r_1, r_2, r_3, r_4, r_5, r_6, p_{H2}, T, F, V_R, X_T, $[Solids]_f$, $[Others]_f$, $[Pr]_f$, $[But]_f$, $[Ac]_f$; (c) 25 model parameters: k_h, u_{max2}, u_{max3}, u_{max4}, u_{max5}, u_{max6}, k_{s2}, k_{s3}, k_{s4}, k_{s5}, k_{s6}, ν_{12}, ν_{15}, ν_{16}, ν_{23}, ν_{24}, ν_{25}, ν_{26}, ν_{35}, ν_{36}, ν_{45}, ν_{46}, ν_{57}, ν_{66}, $\nu 67$; and (d) one constant: R_g. The total number of equations (Ne) is 14, corresponding to 6 ODEs and 8 AEs. So that the degrees of freedom ($DOF = Nv - Ne$) is 35. This means that 35 variables need to be specified. These are the following: the set of 25 parameters, the constant (R_g), plus the following 9 variables T, F, V_R, X_T, $[solids]_f$, $[others]_f$, $[Pr]_f$, $[But]_f$, $[Ac]_f$.
All the parameters correspond to kinetic constants. However, only 8 of them are known from the stoichiometry or from literature (k_h, ν_{35}, ν_{45}, ν_{57}, ν_{67}, ν_{36}, ν_{46}, ν_{66}), so the remaining 17 unknown parameters u_{max2}, u_{max3}, u_{max4}, u_{max5}, k_{s2}, k_{s3}, k_{s4}, k_{s5}, ν_{12}, ν_{23}, ν_{24}, ν_{15}, ν_{25}, u_{max6}, k_{s6}, ν_{16}, ν_{26} must be determined. For this purpose, an optimisation problem and a strategy for solution are defined in order to estimate the unknown kinetic parameters.

◇ **Step 5/6.** *Model data and Model solution*
Having in mind that the original modelling problem of the two-step thermophilic anaerobic digestion of primary or secondary sludge is complex since its kinetic model has 17 unknown parameters, the parameter identification is

divided into three sub-problems. In sub-problem I and II, the determination of the kinetic parameters in the liquid phase is performed, investigating the effect of the pre-treatment step on the activities of the different microbial groups. In Sub-problem III, the determination of the kinetic parameters in the gas phase is performed.

Sub-problem I.
First, only the consumption of acetate, propionate and butyrate are considered under batch conditions, in order to optimise the first set of (six) kinetic parameters u_{max3}, u_{max4}, u_{max5}, k_{s3}, k_{s4}, k_{s5} in digesters A and B separately. For this purpose, four batch kinetic experiments were carried out at constant temperature (55 C) in serum vials, started up with the injection of a relatively small volume of acid (acetate, propionate or butyrate) solution, considering constant biomass concentration (X_T), and using as inoculum anaerobic mixed liquor from:

(a) Batch 1: Digester B fed with non-pretreated primary sludge ($X_T = 1.31068\ g.l^{-1}$),

(b) Batch 2: Digester A fed with pretreated primary sludge at 70 C ($X_T = 0.95146\ g.l^{-1}$),

(c) Batch 3: Digester B fed with non-pretreated secondary sludge ($X_T = 1.4169 g.l^{-1}$),

(d) Batch 4: Digester A fed with pretreated secondary sludge at 70 C ($X_T = 0.7793\ g.l^{-1}$).

Each inoculum was tested with propionate, butyrate or acetate as substrate (i.e., three batch experiments for each inoculum). During the experiments the consumption of the propionate, butyrate or acetate was monitored while the biomass concentrations in the batch digesters were considered constant, and therefore no significant effect on the degradation rate was assumed. The consumption rates are assumed to follow Monod-kinetics:

$$\frac{d\,[Pr]}{dt} = -\frac{u_{\max 3} \cdot [Pr]}{k_{s3} + [Pr]} \cdot X_T \tag{5.100}$$

$$\frac{d[But]}{dt} = -\frac{u_{\max 4} \cdot [But]}{k_{s4} + [But]} \cdot X_T \tag{5.101}$$

$$\frac{d\,[Ac]}{dt} = -\frac{u_{\max 5} \cdot [Ac]}{k_{s5} + [Ac]} \cdot X_T \tag{5.102}$$

Experimental data of $[Pr]$, $[But]$ or $[Ac]$ for each batch kinetic experiment were taken and used to estimate the values of a first set of six kinetic parameters u_{max3}, u_{max4}, u_{max5}, k_{s3}, k_{s4}, k_{s5} that appear in Eqs. 5.100 - 5.102. The optimisation problem for each substrate is stated as:

$$Objective\ 1 = \min \sum_{i=1}^{n} \left\{ \frac{[Pr]_{\exp}(t_i) - [Pr](t_i)}{[Pr]_{\exp}(t_i)} \right\}^2 \tag{5.103}$$

$$Objective\ 2 = \min \sum_{i=1}^{n} \left\{ \frac{[But]_{\exp}(t_i) - [But](t_i)}{[But]_{\exp}(t_i)} \right\}^2 \tag{5.104}$$

$$Objective\ 3 = \min \sum_{i=1}^{n} \left\{ \frac{[Ac]_{\exp}(t_i) - [Ac](t_i)}{[Ac]_{\exp}(t_i)} \right\}^2 \tag{5.105}$$

subject to the ODEs given by 5.100 - 5.102, and with bounds $[Pr](t) \geq 0$, $[But](t) \geq 0$, $[Ac](t) \geq 0$

This problem can be reformulated using the algebraic solutions of Eqs. 5.100 - 5.102 instead of the ODEs, that is,

$$k_{s3} \ln([Pr] / [Pr]_0) + [Pr] - [Pr]_0 + u_{\max 3} \cdot X_T \cdot t = 0 \tag{5.106}$$

$$k_{s4} \ln([But] / [But]_0) + [But] - [But]_0 + u_{\max 4} \cdot X_T \cdot t = 0 \tag{5.107}$$

$$k_{s5} \ln([Ac] / [Ac]_0) + [Ac] - [Ac]_0 + u_{\max 5} \cdot X_T \cdot t = 0 \tag{5.108}$$

Sub-problem II.
Two batch kinetic experiments were carried out at constant temperature (55 C) in serum vials, considering constant biomass concentration (X_T), using:

(a) Batch 5: as inoculum anaerobic mixer liquor from Digester B (thermophilic anaerobic digester fed with non-pretreated primary sludge), and as substrate non-pretreated primary sludge; and

(b) Batch 6: as inoculum anaerobic mixer liquor from Digester B (thermophilic anaerobic digester fed with non-pretreated secondary sludge), and as substrate non-pretreated secondary sludge.

In this case, we consider the complete reaction scheme given in Figure 5.16, in order to optimise a second set with seven unknown parameters. Experimental data of $[Others]$, $[Pr]$, $[But]$ and $[Ac]$ will be used to estimate the values of the kinetic parameters u_{max2}, k_{s2}, ν_{12}, ν_{23}, ν_{24}, ν_{15}, ν_{25} that appear in Eqs. 5.86 - 5.91 (with $F = 0$, since the experiments are in batch mode). Then the optimisation problem is stated as in the following four steps.

Step A. The parameters u_{max2}, k_{s2} and ν_{12} will be optimised using the objective function:

$$Objective\ 4 = \min \sum_{i=1}^{n} \left\{ \frac{[Others]_{\exp}(t_i) - [Others](t_i)}{[Others]_{\exp}(t_i)} \right\}^2 \tag{5.109}$$

subject to Eqs. 5.93-5.94 and

$$\frac{d[Solids]}{dt} = -r_1, \qquad [Solids]_{t=0} = [Solids]_0 \tag{5.110}$$

$$\frac{d[Others]}{dt} = -r_2 + \nu_{12} \cdot r_1, \qquad [Others]_{t=0} = [Others]_0 \tag{5.111}$$

With bounds $[Others](t) \geq 0$.

Step B. The parameter ν_{23} will be optimised using the objective function:

$$Objective\ 5 = \min \sum_{i=1}^{n} \left\{ \frac{[Pr]_{\exp}(t_i) - [Pr](t_i)}{[Pr]_{\exp}(t_i)} \right\}^2 \tag{5.112}$$

subject to the Eqs. 5.93-5.95, 5.110-5.111 and

$$\frac{d[\mathrm{Pr}]}{dt} = -r_3 + \nu_{23} \cdot r_2\,, \qquad [\mathrm{Pr}]_{t=0} = [\mathrm{Pr}]_0 \tag{5.113}$$

With bounds $[Pr](t) \geq 0$

Step C. The parameter $\nu - 24$ will be optimised using the objective function:

$$Objective\ 6 = \min \sum_{i=1}^{n} \left\{ \frac{[But]_{\exp}(t_i) - [But](t_i)}{[But]_{\exp}(t_i)} \right\}^2 \tag{5.114}$$

subject to the Eqs. 5.93-5.96, 5.110-5.111, 5.113 and

$$\frac{d[But]}{dt} = -r_4 + \nu_{24} \cdot r_2, \qquad [But]_{t=0} = [But]_0 \tag{5.115}$$

with bounds $[But](t) \geq 0$.

Step D. The parameter ν_{15} and ν_{25} will be optimised using the objective function:

$$Objective\ 7 = \min \sum_{i=1}^{n} \left\{ \frac{[Ac]_{\exp}(t_i) - [Ac](t_i)}{[Ac]_{\exp}(t_i)} \right\}^2 \tag{5.116}$$

subject to the Eqs. 5.93-5.97, 5.110-5.111, 5.113, 5.115, and

$$\frac{d[Ac]}{dt} = -r_5 + \nu_{45} \cdot r_4 + \nu_{35} \cdot r_3 + \nu_{25} \cdot r_2 + \nu_{15} \cdot r_1, \quad [Ac]_{t=0} = [Ac]_0 \tag{5.117}$$

With bounds $[Ac](t) \geq 0$.

Sub-problem III.

In this sub-problem the goal is to calculate the hydrogen kinetic parameters, this is, the optimisation of the remaining four unknown parameters u_{max6}, k_{s6}, ν_{12}, ν_{16}. For this purpose the two following steps are considered.

1. Step A. Two batch kinetic experiments (by triplicate) were carried out at constant temperature (55 C) in serum vials, started-up with the injection of a relatively small amount of hydrogen, and using as inoculum anaerobic mixed liquor from:

   - (a) Batch 7: Digester B fed with non-pretreated primary sludge ($X_T = 1323 mgl^{-1}$),
   - (b) Batch 8: Digester B fed with non-pretreated secondary sludge ($X_T = 1420.3 mgl^{-1}$).

   The values for hydrogen partial pressure P_{H2} are not direct measurements but they have been estimated from the production of methane during the experiment. Similar to Sub-problem I, we assumed that during the batch experiments the biomass concentration in the batch digesters was constant and the hydrogen consumption rate is assumed to follow Monod-kinetics [Eq. 5.98]. As the hydrogen is the only substrate and the parameter $\nu_{66} = 1$ (due to the stoichiometry), Eq. 5.91 can be rewritten as follows:

   $$\frac{dP_{H2}}{dt} = -r_6 \cdot \left(\frac{V_L \cdot R_g \cdot T}{V_G \cdot 1000} \cdot \gamma \right) \tag{5.118}$$

   The parameters to be estimated are u_{max6}, k_{s6} and the optimisation problem is stated as:

   $$Objective\ 8 = \min \sum_{i=1}^{n} \left\{ \frac{[P_{H_2}]_{\exp}(t_i) - [P_{H_2}]\ (t_i)}{[P_{H_2}]_{\exp}(t_i)} \right\}^2 \tag{5.119}$$

   subject to Eqs. 5.118 and 5.119, and with bounds $[P_{H_2}]\ (t_i) \geq 0$

2. Step B. Similar to Sub-problem II, two batch kinetic experiments were carried out at constant temperature (55 C) in serum vials, considering constant biomass concentration (X_T), using

   - (a) Batch 9: as inoculum anaerobic mixer liquor from Digester B (thermophilic anaerobic digester fed with non-pretreated primary sludge), and as substrate non-pretreated primary sludge; and
   - (b) Batch 10: as inoculum anaerobic mixer liquor from Digester B (thermophilic anaerobic digester fed with non-pretreated secondary sludge), and as substrate non-pretreated secondary sludge.

   Available experimental data of $[C_{H_4}]$ are used to identify the last two kinetic parameters ν_{16}, ν_{26}. The optimisation problem is stated as:

$$Objective\ 9 = \min \sum_{i=1}^{n} \left\{ \frac{[CH_4]_{\exp}(t_i) - [CH_4]\ (t_i)}{[CH_4]_{\exp}(t_i)} \right\}^2 \tag{5.120}$$

subject to Eqs. 5.91-5.92 and 5.98, with bounds $[CH_4]\ (t_i) \geq 0$

5.2.1.1 Step 7/8: Model solution and model validation

Sub-problem I.
The total number of experimental data points for each one of the acids (shown in Figures 5.17-5.20) were: 21 (or 7 experiments by triplicate) for Batch 1, 18 (or 6 experiments by triplicate) for Batch 2, 12 (or 4 experiments by triplicate) for Batch 3, and 18 (or 6 experiments by triplicate) for Batch 4. The optimum parameter values (obtained as described in previous section) are given in Table 5.12, while the dynamic performances of the acid concentrations [by integrating Eqs. 5.100 - 5.102] are shown in Figures 5.17-5.20, where it can be seen that the estimated concentrations are satisfactory and close to the experimental data. The corresponding objective functions [Eqs. 5.103-5.105] are shown in Table 5.13. Based on the parameter standard deviations (given in Table5.12) and on the objective functions evaluations (Table 5.13), it can be conclude that the fitting of the model prediction to the experimental results is quite satisfactory, and therefore the identification of the first set of six kinetic parameters is reliable.

All the model equations given for sub-problem I are then and the set up in ICAS-MoT. The BDF-integration method available in ICAS-MoT together with the SQP optimiser also available in ICAS MoT is used to solve this parameter estimation problem.

Table 5.12: Optimum kinetic parameters for sub-problem I.

Parameter	Batch 1	Batch 2	Batch 3	Batch 4	Units
u_{max3}	0.03630 ± 0.00106	0.05269 ± 0.00116	0.05619 ± 0.00971	0.05370 ± 0.00894	$mmol.g - VS - 1.h^{-1}$
u_{max4}	0.16934 ± 0.01554	0.21362 ± 0.06787	0.17688 ± 0.02596	0.39942 ± 0.02191	$mmol.g - VS - 1.h^{-1}$
u_{max5}	0.43232 ± 0.04511	0.55154 ± 0.04963	0.21832 ± 0.00653	0.23387 ± 0.02197	$mmol.g - VS - 1.h^{-1}$
k_{s3}	0.09744 ± 0.01931	0.84099 ± 0.05524	2.20590 ± 0.79083	0.17561 ± 0.00440	$mmol.l^{-1}$
k_{s4}	1.20382 ± 0.26464	0.22327 ± 0.02912	0.0764 ± 0.04067	2.78658 ± 0.27220	$mmol.l^{-1}$
k_{s5}	1.95080 ± 0.49788	0.52071 ± 0.07979	0.05014 ± 0.01233	1.57174 ± 0.01923	$mmol.l^{-1}$

Table 5.13: Objective functions (Sum of square errors) for Sub-problem I.

	Batch 1	Batch 2	Batch 3	Batch 4
Objective 1	3.076	0.019	0.0176	0.036
Objective 2	0.019	0.008	0.0036	0.0212
Objective 3	1.936	8.653	0.0078	0.0649

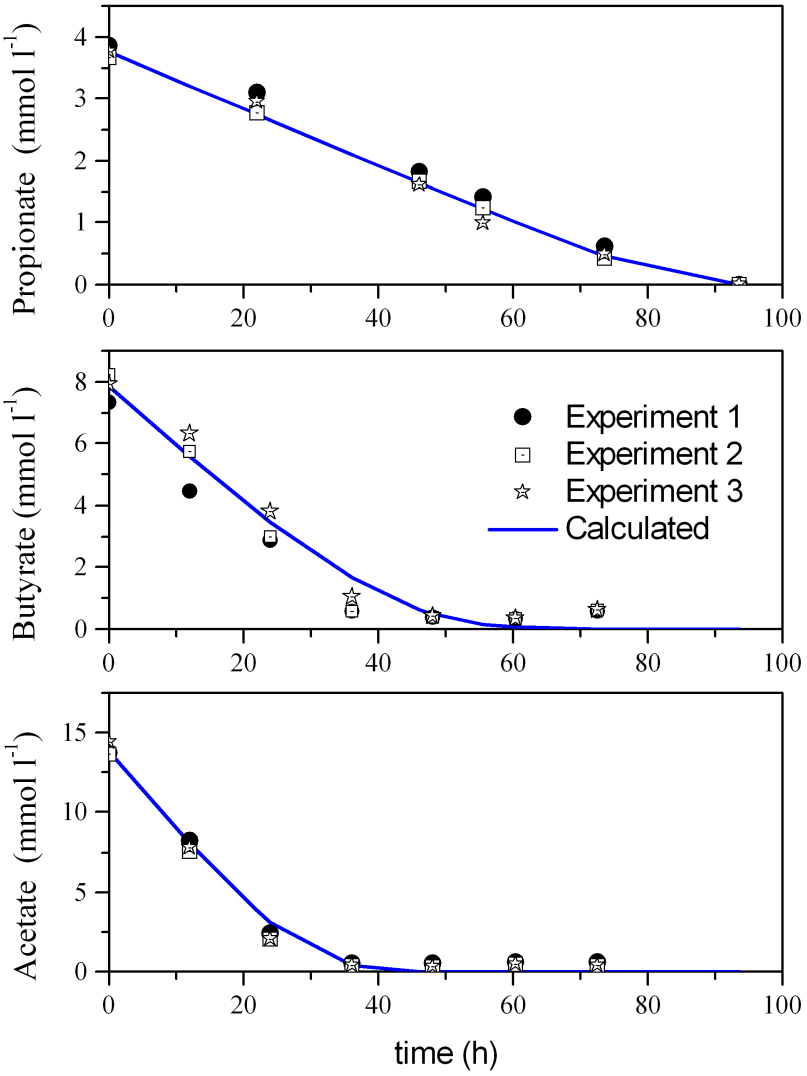


Figure 5.17: Experimental and calculated concentrations for Batch 1

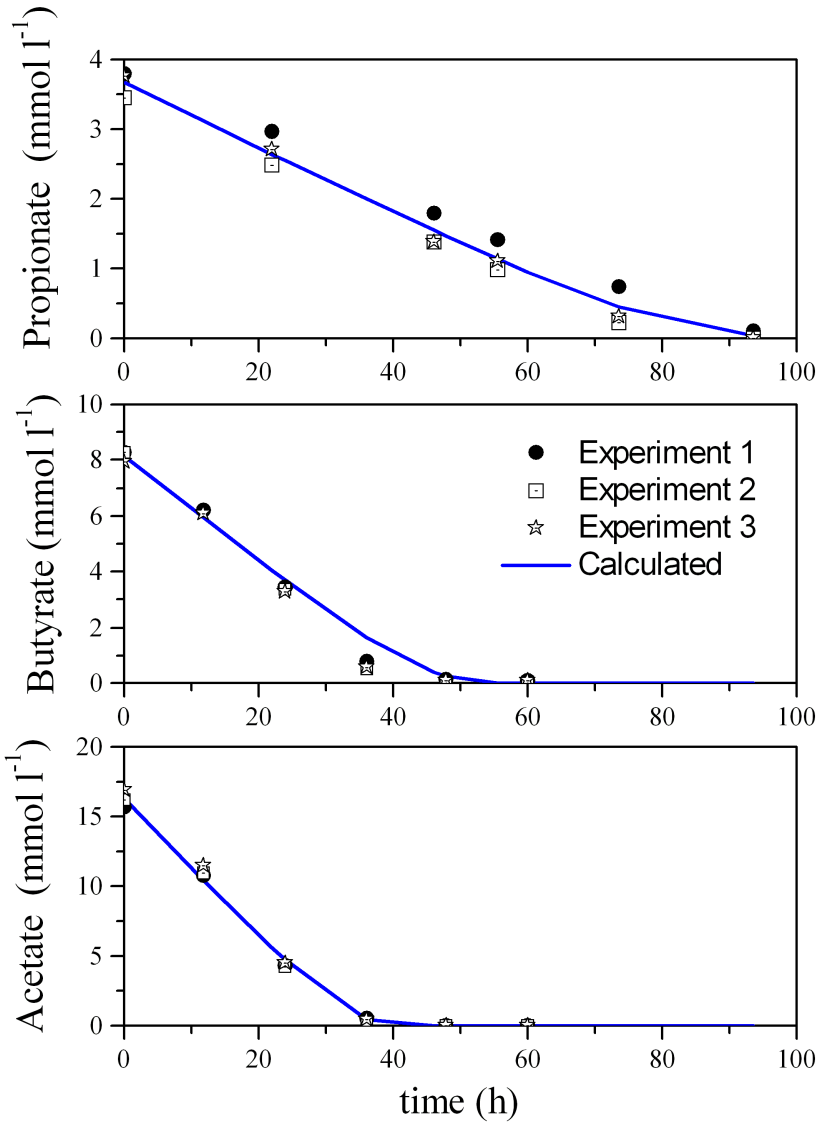


Figure 5.18: Experimental and calculated concentrations for Batch 2

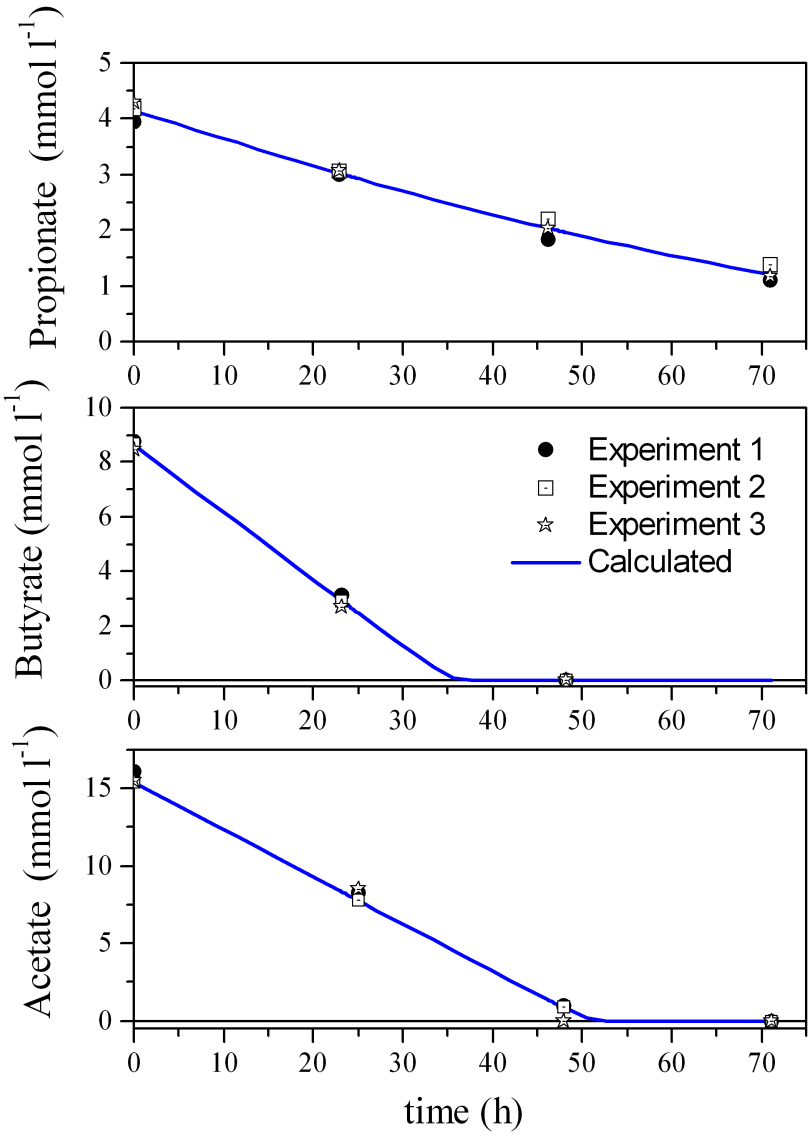


Figure 5.19: Experimental and calculated concentrations for Batch 3

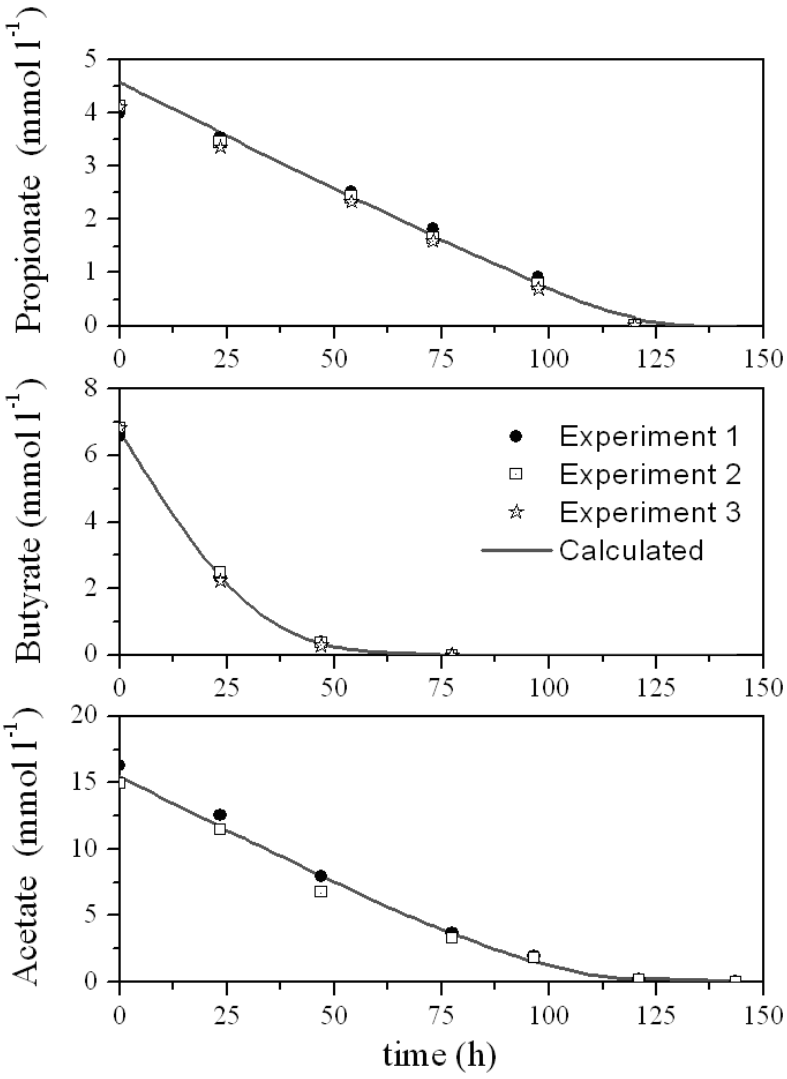


Figure 5.20: Experimental and calculated concentrations for Batch 4

Sub-problem II.
The optimisation was done according the procedure described above. However, a good fit was not obtained for Step 1 because the experimental data for $[Others]$ did not follows the Monod-kinetics. Hence, the optimisation problem for Step 1 was reformulated taking into account the following three modifications:

(i) The concentration of $[Others]$ was identified as the concentration of: (a) non-biodegradable $[Others]$ calculated as $NB[Others]$ and (b) biodegradable others calculated as $(1 - NB)[Others]$, where NB is the percentage of the non-biodegradable $[Others]$. Then its mass balance for non-biodegradable others is given by

$$\frac{dS}{dt} = D \cdot NB \cdot [Others]_{in} - D \cdot [Others]_{eff} + v_{12} \cdot NB \cdot K_h \cdot [Solids] \quad (5.121)$$

which at steady state gives:

$$0 = D \cdot NB \cdot [Others]_{in} - D \cdot [Others]_{eff} + v_{12} \cdot NB \cdot K_h \cdot [Solids] \quad (5.122)$$

or else

$$NB = \frac{D \cdot [Others]_{eff}}{D \cdot [Others]_{in} + v_{12} \cdot K_h \cdot [Solids]_{ss}} \quad (5.123)$$

(ii) The reaction rate was considered as first order (instead of Monod-type), this is:

$$r_2 = k_2 \cdot (1 - NB) \cdot [Others] \cdot X_T \quad (5.124)$$

(iii) The parameter ν_{12} was identified from another independent set of experimental data, so that the parameters to be identified are now K_h, k_2 instead of $u_{max}2$, $k_{s}20$, $\nu_1 2$. On the other hand, it was necessary to optimise the parameters u_{max5} and k_{s5} together with ν_{15} and ν_{25} in order to get a good fit in **Step D**. Summarizing, **Step 1** to **Step 4** were reformulated as follows:

- Step A. The parameters K_h and k_2 are optimised using the objective function given by Eq. 5.109, subject to the Eqs. 5.110-5.111, 5.93, and 5.123-5.124, with bounds $[Others](t) \geq 0$.
- Step B. The parameter ν_{23} is optimised using the objective function given by Eq. 5.112, subject to the Eqs. 5.110-5.111, 5.113, 5.93, 5.123-5.124 and 5.95, with bounds $[Pr](t) \geq 0$.

- Step C. The parameter ν_{24} is optimised using the objective function given by Eq. 5.114, subject to the Eqs. 5.110-5.111, 5.113, 5.115, 5.93, 5.123-5.124 and 5.95-5.96, with bounds $[But](t) \geq 0$.
- Step D. The parameters u_{max5}, k_{s5}, ν_{15} and ν_{25} are optimised using the objective function given by Eq. 5.116, subject to the Eqs. 5.110-5.111, 5.113, 5.115, 5.117, 5.93, 5.123-5.124 and 5.95-5.97, with bounds $[Others](t) \geq 0$.

Then, sub-problem II is solved using this reformulation. The total number of experimental data points used for each one of the acids (shown in Figures 5.21 and 5.22) were: 21 (or 7 experiments by triplicate) for Batch 5, and 39 (or 13 experiments by triplicate) for Batch 6. The known parameters to solve this sub-problem are given in Table 5.14, the optimum parameter values are reported in Table 5.15, the corresponding objective functions [Eqs. 5.109, 5.112, 5.114 and 5.116] are shown in Table 5.16, and the dynamic performances of the acid concentrations [by integrating Eqs. Eqs. 5.110-5.111, 5.113, 5.115, 5.117 together with 5.123-5.124 and 5.95-5.97] are shown in Figures 5.21 and 5.22.

Table 5.14: Known parameters for sub-problem II

Parameter	Batch 5	Batch 6	Units	Reference
$[Others]_{in}$	1339	772	$mg - COD.l^{-1}$	Experimental
$[Others]_{eff}$	542	368	$mg - COD.l^{-1}$	Experimental
$[Solids]_{ss}$	5150	5105	$mg - VSS.l^{-1}$	Experimental
D	0.00278	0.06667	$h - 1$	Experimental
X_T	1286	1276	$mg - VS.l^{-1}$	Experimental
$[Solids]_0$	8576	5364	$mg - COD.l^{-1}$	Experimental
$\nu : 12$	1.7855	1.6742	$mg - CODOthers/mgVSS$	Experimental
ν_{35}	0.57143	0.57143	$mg - CODAc/mgPr$	Stoichiometry
ν_{45}	0.8	0.8	$mg - CODAc/mgBut$	Stoichiometry
u_{max3}	0.00406	0.00629	$mg - CODmg - VS^{-1}.h^{-1}$	Sub-prob. I
k_{s3}	10.91328	247.0608	$mg - COD.l^{-1}$	Sub-prob. I
u_{max4}	0.02709	0.0283	$mg - COD.mg - VS^{-1}.h^{-1}$	Sub-prob. I
k_{s4}	192.6112	12.224	$mg - COD.l^{-1}$	Sub-prob. I
u^*_{max5}	0.02767	0.01397	$mg - COD.mg - VS^{-1}.h^{-1}$	Sub-prob. I
k^*_{s5}	124.8512	3.20896	$mg - COD.l^{-1}$	Sub-prob. I

Sub-problem III.
This sub-problem is solved as defined previously. The total number of experimental data points available for Step A (shown in Figures 5.23 and 5.24) were: 33 (or 11 experiments by triplicate) for Batch 7, and 6 (or 6 experiments by triplicate) for Batch 8; while for Step B (shown in Figures 5.25 and 5.26) were: 24 (or 8 experiments by triplicate) for Batch 9, and 39 (or 13 experiments by triplicate) for Batch 10. The known parameters for both batches to solve Step A: $V_L = 0.0204\ l$, $V_G = 0.0376\ l$ and $g = 0.0625\ mmolH_2.mg^{-1}CODH_2$; while the known parameters for Step B are given in Table 5.17. The optimum parameter values are reported in Table 5.17, the corresponding objective functions [Eqs. 5.119 and 5.120] are reported in Table 5.19, and the dynamic

Table 5.15: Optimum kinetic parameters for Sub-problem II.

Parameter	Batch 5	Batch 6	Units	Step
K_h	$9.0704\text{x}10^{-3} \pm 2.7591\text{x}10^{-4}$	$6.48460\text{x}10^{-3} \pm 1.33915\text{x}10^{-4}$	h^{-1}	1
k_2	$8.9580\text{x}10^{-5} \pm 1.8255\text{x}10^{-6}$	$4.02177\text{x}10^{-5} \pm 6.24463\text{x}10^{-7}$	$lg - VS^{-1}h^{-1}$	1
ν_{23}	0.06427 ± 0.0015789	0.12029 ± 0.00368	$mg - CODPr/mg - CODOthers$	2
ν_{24}	0.13603 ± 0.0065041	0.69998 ± 0.00944	$mg - CODBut/mg - CODOthers$	3
ν_{25}	0.00000 ± 0.00000	0.00000 ± 0.00000	$mg - CODAc/mg - CODOthers$	4
ν_{15}	0.56320 ± 0.12814	3.29811 ± 0.05030	$mg - CODAc/mg - VSS$	4
u_{max5}	0.05039 ± 0.00795	0.09774 ± 0.00303	$mg - COD.mg - VS{-}1.h{-}1$	4
k_{s5}	34.21409 ± 7.61346	13.36953 ± 0.68349	$mg - COD.l{-}1$	4

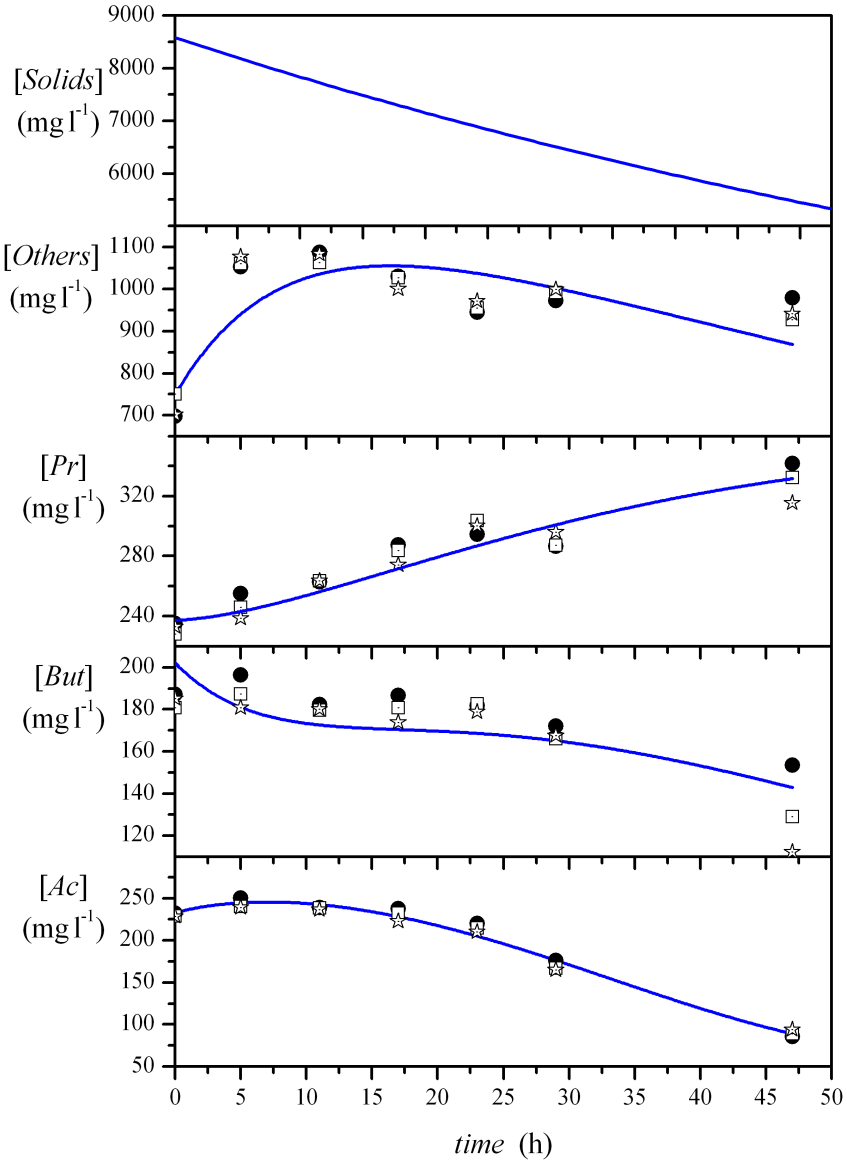


Figure 5.21: Experimental and calculated concentrations for Batch 5.

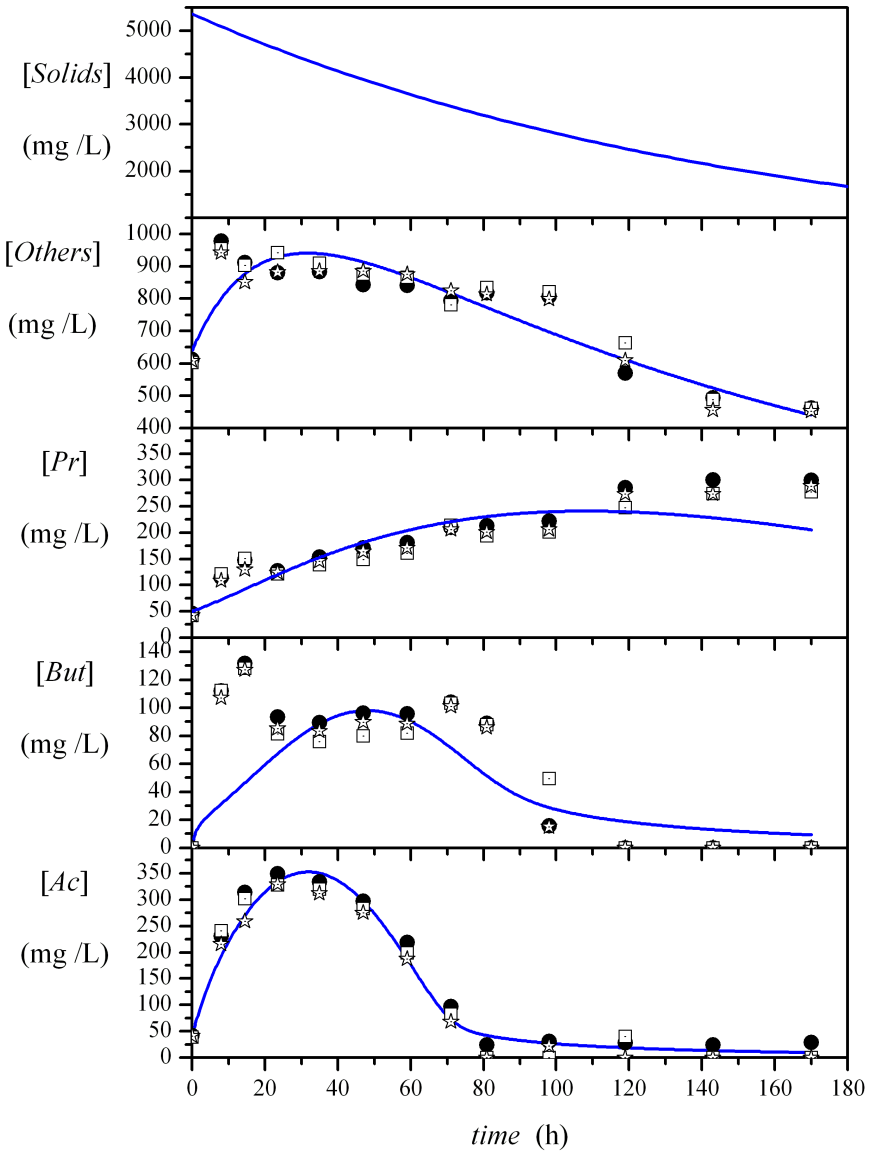


Figure 5.22: Experimental and calculated concentrations for Batch 6.

Table 5.16: Objective functions for parameters for Sub-problem II.

	Batch 5	Batch 6
Objective 4	0.03338	0.07735
Objective 5	0.00669	0.52359
Objective 6	0.03999	1.76545
Objective 7	0.00479	0.04075

performances of the hydrogen and methane concentrations are shown in Figures 5.23-5.26.

Table 5.17: Known parameters for Step B of sub-problem II.

Parameter	Value	Units	Reference
V_L	0.108	l	Experimental
V_G	0.219	l	Experimental
R_g	0.082057	$l.atm.mole-1.K-1$	Literature
T	328	K	Experimental
g	0.0625	$mmoleH_2.mgCODH_2$	Stoichiometry
d	0.015625	$mmoleCH_a.mgCODCH_4$	Stoichiometry
ν_{66}	1		Stoichiometry
ν_{46}	32/160	$mgCODH_2permgCODBut$	Stoichiometry
ν_{36}	48/112	$mgCODH_2permgCODPr$	Stoichiometry
ν_{57}	64/64	$mgCODCH_4permgCODAc$	Stoichiometry
ν_{67}	64/64	$mgCODCH_4permgCODH_2$	Stoichiometry

Table 5.18: Optimum kinetic parameters for sub-problem III.

Parameter	Batch 5	Batch 6	Units	Step
u_{max6}	0.01706 ±0.00290	0.07807 ± 0.00450	$atm.l.mgVS^{-1}.h^{-1}$	A
$k_{s6}4$	0.01758±0.01023	0.01758±0.01023	atm	A
ν_{26}	0.50218 ± 0.00124	0.16696 ± 0.00052		B
ν_{16}	0.50125 ± 0.00072	0.0000 ± 0.0000		B

According to the results of all three sub-problems, it can be concluded that the fitting of the kinetic parameters is satisfactory and reliable. Moreover, the parameter ν_{25} was found to be zero for both the primary and the secondary sludge, meaning that the fermentation of *Others* only produces butyrate and propionate, but not acetate as stated in the original kinetic mechanism (Figure 5.16). In this way the kinetic mechanism is corrected as shown in Figure 5.23. Also the kinetic reaction rate for *Others* (i.e. r_2) was found to follow a first order kinetics [Eq. 5.124] instead of Monod-kinetics [Eq. 5.94].

Case summary

Thermal pre-treatment of primary and secondary sludge at 70 C enhances the

Table 5.19: Objective functions for parameters for sub-problem III

	Batch 7	Batch 8	Batch 7	Batch 8
Objective 8		NA	-	-
Objective 9	-	-	1.44853	1.246278

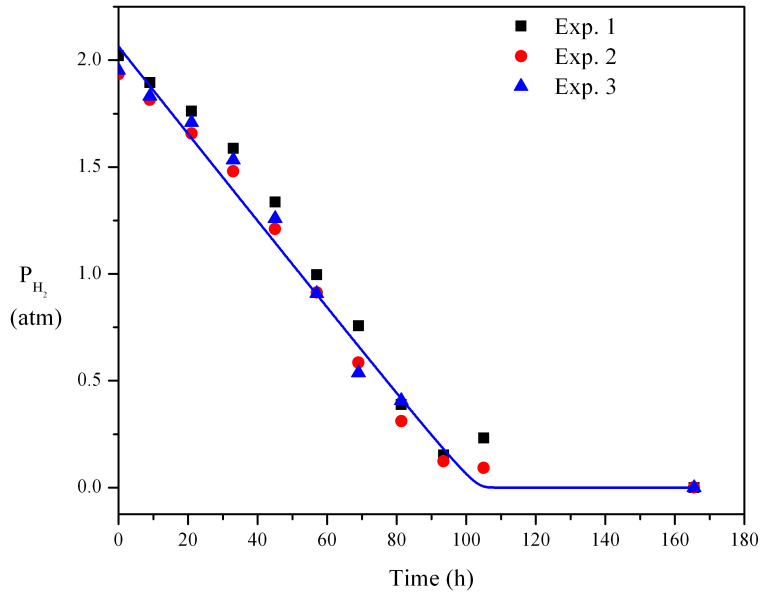


Figure 5.23: Experimental and calculated concentration of H_2 for Batch 7.

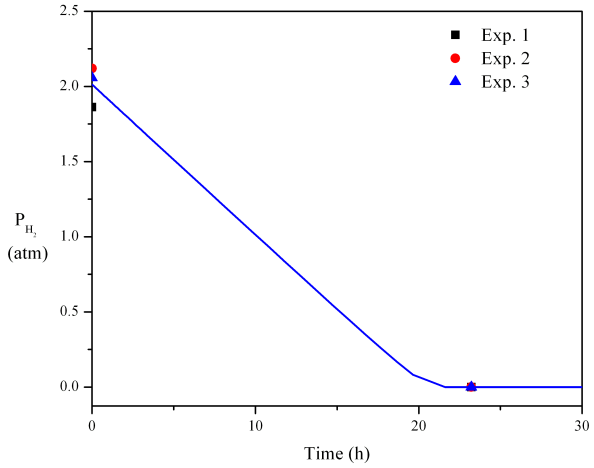


Figure 5.24: Experimental and calculated concentration of H_2 for Batch 8.

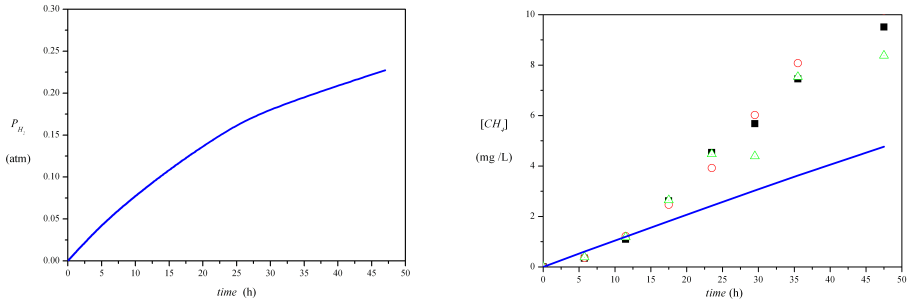


Figure 5.25: Experimental and calculated concentration of hydrogen and methane for Batch 9.

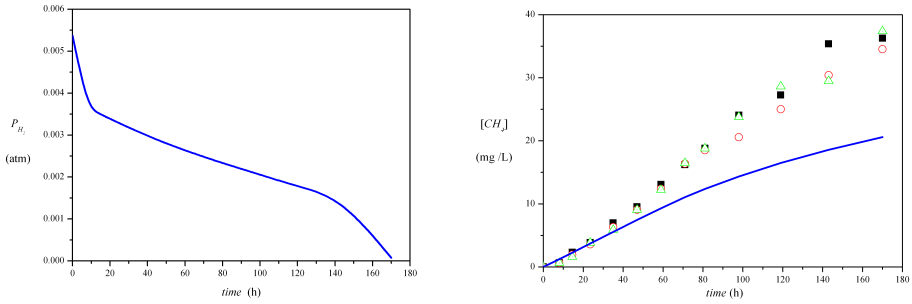


Figure 5.26: Experimental and calculated concentration of hydrogen and methane for Batch 10.

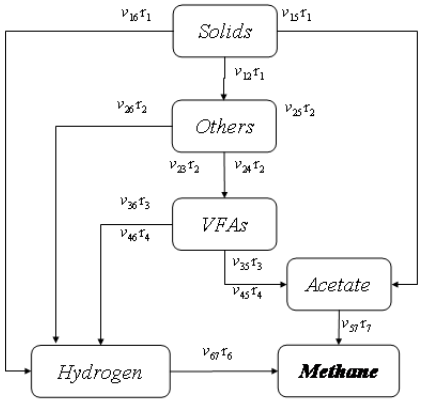


Figure 5.27: Corrected kinetic mechanism for anaerobic bio-conversion of organic matter to methane.

removal of organic matter and the methane production during the subsequent anaerobic digestion step at 55 C and greatly contributes to the destruction of pathogens present in primary sludge. It results in enhanced microbial activities of the subsequent anaerobic step allowing the operation of treatment units at shorter hydraulic retention time. A strategy for the kinetic characterization was proposed, such that the final identified kinetic model is able to adequately and reliably simulate the thermophilic anaerobic digestion of primary and secondary sludge. The establishment of a modelling based framework to systematically optimise the performance of the anaerobic digestion process and the identification of the main process variables, parameters and operation conditions, as well as for the efficient designing of experiments was aided by ICAS-MoT. This case study formed the basis for implementation and testing of two new important options (tools) of ICAS-MoT: The dynamic optimisation tool for the kinetic parameter identification based on experimental data, and the use of statistics information reported by ICAS-MoT to validate and discriminate the suggested kinetic models.

5.2.2 Process Optimisation: Short-Path evaporator.

An important stage in the design process for many chemical products is its manufacture where, for a class of chemical products that may be thermally unstable (such as, drugs, insecticides, flavours/fragrances, etc.), the purification step plays a major role. Short-path evaporation is usually a safe method, suitable for separation and purification of thermally unstable materials whose design and analysis can be efficiently performed through reliable model-based techniques. This section presents a generalized model for short-path evaporation and highlights its development, implementation and solution through our computer-aided modelling framework (ICAS-MoT), which allows the use of systematic simulation strategies for various types of design/analysis problems.

◇ **Step 1.***System description*
Conventional distillation is one of the oldest methods to separate liquid or molten substances. However, it is often not recommended for substances that can be degraded under distillation temperatures, such as vitamins, insecticides, drugs and flavours/fragrances. Short-path evaporation is a separation technique used as an alternative in various processes of the chemical, pharmaceutical, fragrance and food industry. It is normally a safe method suitable for separation and purification of thermally unstable materials, through a small distance between the evaporator and the condenser, and characterized by low temperatures, short residence times of the distilled liquid on the thermally exposed surface and sufficiently low pressure in the distillation gap (space between evaporator and condenser). This method is also called molecular distillation because the vapour path is unobstructed, and the condenser is separated from the evaporator by a distance less than the mean free path of the evaporating molecule. Therefore, the modelling, design and analysis of short-path evaporation (or molecular distillation) are important elements in many chemical product engineering problems. The short-path evaporation (separation) process may not function properly if the temperature conditions on the condensation surface do not enable total condensation. In this case, process condensation becomes the limiting factor of the whole equipment. Moreover, information about the film surface temperature on the condensation surface is important to determine yield and purity of the distilled product, as well as to define the evaporator design (i.e., the feed position and the evaporator geometry). However, direct measurement of the temperature profile in the film of the condensate is extremely difficult. So a key issue is the building of an appropriate model that can describe the separation process as a function of the film profiles (concentration, temperature and velocity), which can be useful for operation analysis and process design. Several authors have previously tackled the modelling of the short-path evaporator. Most of the reported models have been developed for binary mixtures, some have been tested with experimental data (Kawala (1976); Nguyen, and Goffic (1997); Lutišan, Cvengroš, and Micov

(2002)). Other works have concentrated on specific design issues. For instance, (Kawala 1976) studied (experimentally) the effect of anisotropic properties on the rate of evaporation of binary mixtures. (Kawala, and Stephan 1989) simulated the process in a falling film evaporator with adiabatic regime. (Batistella and Maciel 1996) compared the performance of centrifugal and falling film evaporators for binary mixtures, using the model developed by (Kawala et al. 1989). (Nguyen et al. 1997) developed a model for the separation of binary mixtures under the assumption of no temperature change in the liquid flow, thereby neglecting the heat balance. Micov, Lutišan, and Cvengroš (1997) developed a model for binary mixtures, taking into account the mass transfer in the vapour phase, the film flow in the gravity film, the diffusion and energy balance, but the effect of collisions in the distillation gap and of the pressure were neglected and experimental verification was not reported. (Cvengroš, Lutišan, and Micov 2000) employed the same model to study the effect of temperature on the evaporation efficiency of a feed liquid containing only one component. (Cvengroš, Micov, and Lutišan 2000) also studied a mode of operation of the short-path evaporator with a divided condenser that can be used for fractioning and recycling a portion of the distillate. Recently, (Lutišan et al. 2002) investigated the effect of hydrodynamic conditions (i.e. turbulent or laminar regime). Their study was based on the model developed by (Micov et al. 1997); and was tested only for a binary mixture obtaining good qualitative agreement between experimental and theoretical results, but not good quantitative results. Therefore, a generalized model covering a wide range of multi-component mixtures as well as operational and configurational options would be a valuable addition for the systematic design and analysis of the separation and/or purification of selected chemical products through short-path evaporation operations. Use of ICAS-MoT, through which particular forms of the generalized model can be created and solved for various design/analysis problems, would be an additional benefit.

The objective here is to present a generalized short-path evaporation model together with a corresponding systematic simulation strategy and analysis of simulation results, with particular emphasis in analysis and design issues such as industrial process operation validation, sensitivity analysis, verification/design of operational conditions for a desired separation, and, improving the yield and purity of the desired chemical product. The modelling and simulation should also help to understand how the feed concentration, the feed temperature, the heating surface, the system pressure and evaporator dimensions (length and gap) affect the film temperature profile, the film thickness, the evaporation rate and the evaporation yield. As starting point, a generalized two-dimensional steady-state mathematical model is developed based on mass, heat and momentum balances, resulting in a set of PDAEs.

Figure 5.28 shows the tools that should be used to implement and solve the problem described above using ICAS-MoT. This problem deals with a set of PDAE systems. These equations are discretized internally and the (DAE)

system generated is solved used the available BDF method in ICAS-MoT to perform the integration, .

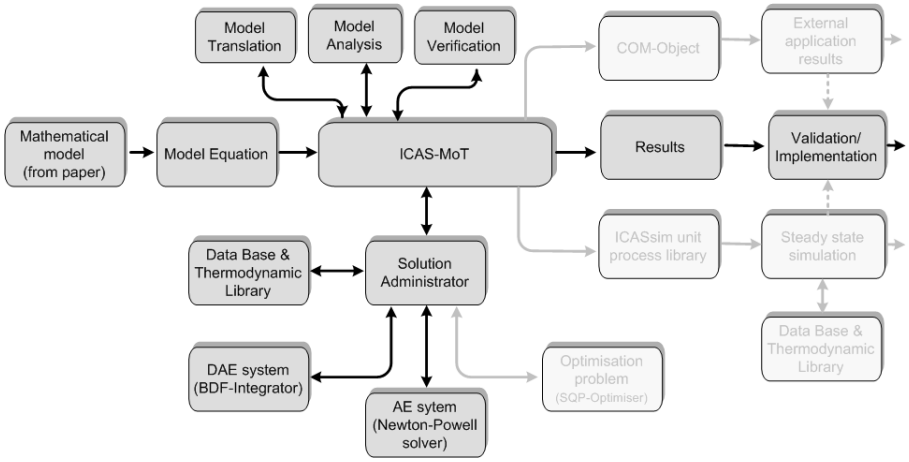


Figure 5.28: Work-Flow and tools used from ICAS-MoT for distributed model simulation (case study 2)

The work-flow that should be followed to solve the problem is highlighted with dark grey colour, whereas the blocks in light grey means that the path is inactivate for problems such as the one tackled here. This work-flow involves the main steps to set up the model and the DAE integrator solver to solve it.

◇ **Step 2.***Problem definition*
The short-path evaporator consists of a cylindrical body surrounded by a cylinder, which provides the surfaces for the evaporation film and the condensation film (Figure 5.29). The liquid solution to be processed (purified) is fed in the evaporation wall by means of a suitable pumping system. The evaporation and condensation surfaces are kept at constant temperatures T_{w1} and T_{w2}, respectively. Due to the high vacuum inside the separator, a falling non-boiling film is formed, the concentration and temperature profiles (see Figure 5.30b) of the most volatile compounds decrease in z and y directions, and the velocity profile (see Figure 5.30c) behaves like a laminar flow with a smooth film surface.

The short residence time of the liquid on the evaporating cylinder is guaranteed by distributing the liquid in the form of a thin film of even consistency, while the high vacuum reduces the distillation temperature. Thus, the combination of having a small gap between evaporator and condenser with high vacuum results in a specific mass transfer mechanism: the transfer resistance is reduced, and the distillation rate is increased. Moreover, in one-dimensional distillation vapour, molecules emanate from the hot surface towards the cooler condenser situated directly opposite. In addition, some molecules leave the condenser surface steam in the reverse direction, towards the evaporator. Col-

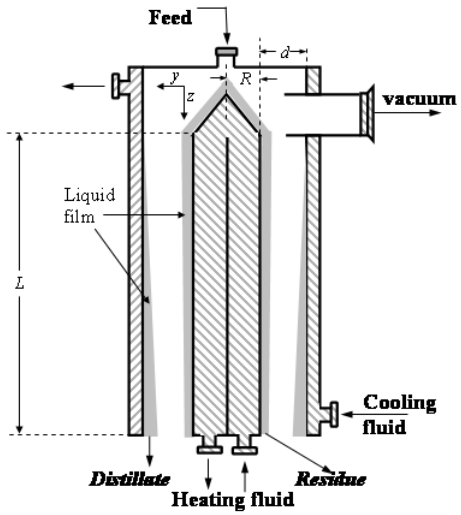
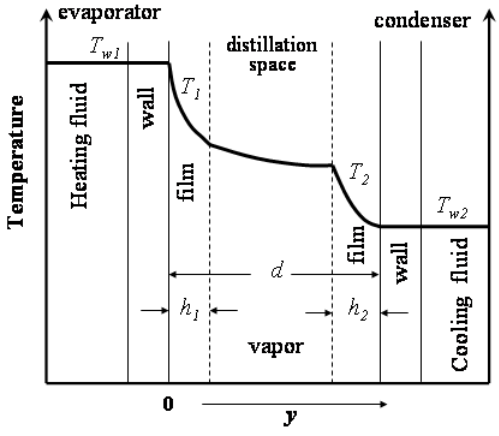


Figure 5.29: Scheme of the short-path evaporator.



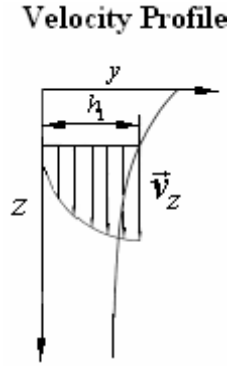


Figure 5.30: (b) Scheme of temperature profile and (c) Scheme of the velocity profile and film shape on the short-path evaporator.

lisions among vapour molecules (anisotropic properties), or with inert gases that are dissolved in liquid feed, may cause a certain fraction of the evaporated fluid to be deflected backwards in the direction of the evaporator surface (Kawala et al. 1989). Based on experimental results, (Kawala 1976) found that the degree of anisotropy of the vapour phase (computed using the dimensions of the evaporation and condensation walls) can be taken into account to calculate the effective rate of evaporation, assuming that the anisotropic properties of the vaporized molecules fade if the number of collisions is more than two. Therefore, the number of collisions can be modified by changing the short-path evaporator geometry, thereby improving the evaporation rate and separation efficiency. Therefore, the combined processes of evaporation-condensation and anisotropic properties of the vapour contribute to a high quality separation and purification.

◇ **Step 3.***Model construction*
The model developed by (Micov et al. 1997) has been taken as the starting point. This model considers the following fundamental issues: (a) separation of binary mixtures, (b) mass transfer in the vapour phase, (c) film flow in the gravity film, (d) an equation for diffusion, and (e) thermal balance. The new generalized model has been developed by considering the following additional issues: (i) extension to handling of multi-component mixtures, (ii) account for the degree of anisotropy of the vapour phase in the space between the evaporator and the condenser, (iii) correct the rate of evaporation due to effects of the vacuum condition, and (iv) introduce models for calculation of component activity coefficients in the liquid phase. The developed generalized two-dimensional steady-state model for short path evaporation makes the following assumptions:

$\mathcal{A}_1$. The process is at steady-state.

$\mathcal{A}_2$. The liquid films on the evaporation and condensation walls are much thinner than the corresponding cylinder diameters.

$\mathcal{A}_3$. Rectangular coordinates are used.

$\mathcal{A}_4$. The liquids are Newtonian.

$\mathcal{A}_5$. The flow in the vertical direction is laminar.

$\mathcal{A}_6$. Re-evaporation and splashing phenomena are neglected.

$\mathcal{A}_7$. Operation occurs far from the extremities of the evaporator (i.e. for a fully developed flow).

$\mathcal{A}_8$. There is no diffusion in the axial direction(z) and the flow in y direction is neglected.

Based on the above assumptions and applying momentum, energy and mass balances (see Figure 5.31) for both evaporation and condensation films, the mathematical model is derived next.

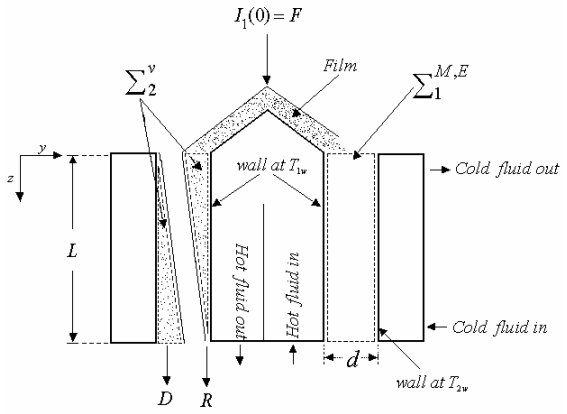


Figure 5.31: Balance volumes for short-path evaporator. $\sum_1^{M,E}$ is the balance volume (1) for mass (M) and Energy (E) related to evaporator and condenser, and $\sum_2^v$ is the balance volume (2) for momentum (v) related to the film.

Momentum balance

In most cases of short-path evaporation, the evaporating liquid is highly viscous and hence the corresponding Reynolds numbers are small. The Navier-Stokes equation (at steady state) for laminar flow regime describes the velocity profile (see Figure 5.30(c)) of falling film

$$\nu \frac{\partial^2 v\,(y,z)}{\partial y^2} = -g \tag{5.125}$$

where y and z are the rectangular coordinates, v is the velocity and g is the gravitational constant. Eq. (5.125) has the following boundary conditions

$$v\left(0,z\right)=0, \qquad v\left(y,z\right)=v_{\max} \tag{5.126}$$

Rate of Evaporation
The rate of evaporation is obtained from the continuity equation in terms of flow rate (I_i) for each component i.

$$\frac{\partial I_i\left(z\right)}{\partial z}=-2\pi\cdot R\cdot k_i, \qquad i=1,\ldots,N; \qquad I_i\left(0\right)=I_{i,o} \tag{5.127}$$

Where, the effective rate of evaporation of each component (k_i) is calculated through a modified Langmuir-Knudsen equation (Kawala et al. 1989)

$$k_i=\frac{\gamma_i p_i^{vap}}{\sqrt{2\pi R_g M_i T_s\left(z\right)}}\left(\frac{P}{P_{ref}}\right)\left\{1-\left(1-F\right)\left[1-e^{d/(\kappa\beta)}\right]^n\right\}, \qquad i=1,\ldots,N \tag{5.128}$$

Eq. 5.128 contains a factor (P/P_{ref}) for correcting the vacuum pressure, as well as a correction that takes into account the anisotropic properties of the vapour, where β is the mean path of vapour molecule, d is the distillation gap, n is the number of intermolecular collision, F is the surface ratio and κ is the degree of anisotropy of the vapour phase in the space between the evaporator and the condenser. (Kawala 1976) reported that the best agreement between experiments and model is obtained for $n=5$. Note that F and κ are calculated through the following equations (Kawala et al. 1989)

$$F=\frac{A_k}{A_k+A_V} \tag{5.129}$$

$$\log\kappa=0.2F+1.38\left(F+0.1\right)^4 \tag{5.130}$$

Where, A_k and A_v are the condensation and evaporation areas, respectively. The effective rate of evaporation [Eq. 5.128] also depends on some mixture properties (activity coefficient γ_i, vapour pressure p_i^{vap} and molecular weight M_i of each compound) as well as on design parameters (the radius of the evaporator inside cylinder R and the surface temperature T_s).

Energy Balance

The temperature (T) profile in the falling film is given by the equation

$$v\left(y,z\right)\frac{\partial T\left(y,z\right)}{\partial z}=\frac{\lambda}{\rho Cp}\left[\frac{\partial^2 T\left(y,z\right)}{\partial y^2}+\frac{\partial^2 T\left(y,z\right)}{\partial z^2}\right] \tag{5.131}$$

With boundary conditions at $z = 0$ (i.e. at the feed position the temperature corresponds to the liquid feed temperature), at $y = 0$ (i.e. at the evaporation surface, the temperature corresponds to the wall evaporation temperature) and $y = h_1$ (i.e. the heat flux from the liquid film surface is given by the evaporation heat ΔH^{vap} and the effective net rate of evaporation k):

$$T\left(y,0\right)=T_F,\qquad T\left(0,z\right)=T_{w1},\qquad \lambda\frac{\partial T\left(y,z\right)}{\partial y}\bigg|_{y=h_1}=\Delta H^{vap}\cdot k \tag{5.132}$$

$$k=\sum\nolimits_{i=1}^{N}\left(\frac{C_i k_i}{\sum_{k=1}^{N}C_k}\right) \tag{5.133}$$

Where λ, ρ, C_p, ΔH^{vap} are the thermal conductivity, density, thermal capacity and heat of evaporation of the multi-component mixture, respectively; and x_i is the mole fraction of the i-th component.

Mass Balance

The composition (C_i) profiles for each component are calculated from the diffusion equation

$$v\left(y,z\right)\frac{\partial C_i\left(y,z\right)}{\partial z}=D_i\left[\frac{\partial^2 C_i\left(y,z\right)}{\partial y^2}+\frac{\partial^2 C_i\left(y,z\right)}{\partial z^2}\right],\qquad i=1,\ldots,N \tag{5.134}$$

Where D_i is the (constant) diffusion coefficient for the i-th component, and N is the total number of components. The boundary conditions for Eq. 5.134 are

$$C_i\left(y,0\right)=C_{i,o},\qquad \frac{\partial C_i\left(0,z\right)}{\partial y}=0,\qquad D_i\frac{\partial C_i\left(y,z\right)}{\partial y}\bigg|_{y=h_1}=I_i\left(z\right) \tag{5.135}$$

Film thickness

Finally, an important variable of interest is the film thickness (h_1) along the evaporator height that is calculated as follows (Cvengroš, Lutišan, and Micov 2000)

$$h_1\left(z\right)=\sqrt[3]{\frac{3\nu}{2\pi\cdot R\cdot g\cdot c}I\left(z\right)} \tag{5.136}$$

$$c=\sum_{i=1}^{N}C_i\left(z\right) \tag{5.137}$$

$$I\left(z\right)=\sum_{i=1}^{N}I_i\left(z\right) \tag{5.138}$$

Where, $\upsilon=\eta/\rho$ is the kinematics viscosity of the multi-component mixture.

The activity coefficients γ_i are calculated using the original UNIFAC approach (Fredenslund et al. (1975)), the diffusion coefficients D_i are calculated according to (Reddy and Doraiswamy 1967), while the physicochemical properties (i.e. λ_i, ρ_i, Cp_i, ΔH_i^{vap}, P_i^{vap}, η_i) are calculated through temperature-dependent relationships. The constitutive models used for these properties are given by Eqs(5.139)-(5.144)(see Table 5.21).

The short-path evaporation model represented by Eqs. 5.125-5.138 can be considered a generalized model since: (a) it is valid for multi-component mixtures; (b) it considers mass and/or energy balance; (c) it accounts for processes

in liquid films on both evaporator and condenser as well as in the gas phase in the distillation gap, so that the location of the evaporation and condensation surfaces can be interchanged; and (d) the arranged set of balance equations enables the study of various operational scenarios in the molecular evaporator to be modelled (i.e., effect of feed temperature and flow rate, column pressure, etc.) and of the influence of equipment parameters of the short-path evaporator to be analysed.

◇ **Step 4.***Model analysis*
The use of ICAS-MoT starts with the import/creation of the model, followed by model translation and analysis. After model translation (converting the model equations from text-mode to a format the computer understands), the variables are classified as explicit, parameter, unknown, known or dependent; then the ordinary and partial differential equations (if they are found in the model) are paired to their dependent variables by the user. Any change in the variable/equation classification is immediately reflected (and automatically done by the system) in the incidence matrix, degrees of freedom (DOF) and if singularity condition of the incidence matrix is obtained (unless the equation system has index > 1, a well posed problem satisfying the DOF needs to be non-singular). The DAE solver in ICAS-MoT only solves index 0-1 problems. Higher index problems are identified by ICAS-MoT during model analysis and reformulation or index reduction through differentiation is proposed as alternative before the solution steps.
The short-path evaporator model described above, consisting of Eqs. 5.125-5.144, is a PDAE system with $(7 + 9N)$ total equations classified as: $(2 + N)$ PDEs [Eqs. 5.125, 5.131 and 5.134], N ODEs [Eqs. 5.127] and $(5 + 7N)$ AEs [Eqs. 5.128-5.130, 5.136-5.138, and 5.139-5.144]. There are $(19 + 14N)$ total variables, which are classified (see Table 5.20) as: $(6 + 2N)$ known, $(6 + 9N)$ model parameter, $(2 + 2N)$ dependent and $(5 + N)$ explicit variables. And so, the DOF is equal to $12 + 11N$, meaning that the variables that need to be specified are: 3 variables fixed by the problem (constants - R_g, g and p), $3 + (3 + 2N)$ variables fixed by the system (equipment dimensions - R, d and L; and, operating conditions - T_{w1}, P, T_F, C_{io}, I_{io}), 3 variables fixed by the property model (model parameters - n, b and P_{ref}) and $9N$ adjustable (or regressed) variables fixed by the pure compounds (property model parameters - M_{wi}, λ_i, ρ_i, Cp_i, ΔH_i^{vap}, D_i, P_i^{vap}, η_i, γ_i). In fact, the operation conditions are the variables that the user will be interested in playing with during the simulation in order to see the effect of their variation on the separation efficiency, which in turn is defined by the outlet variables that must be calculated by the model. By changing the classification of known variables as unknown and vice versa, different simulation problems for design/analysis can be generated, as long as the DOF is not violated and the non-singularity condition is satisfied. For instance, to optimise the separation efficiency, the desired values of the outlet flows can be fixed as known so that now the unknown variables could be: (a) the equipment dimensions (i.e. R, d, L) or (b) the operation conditions (i.e.

P, T_F, T_{w1}). This reformulation procedure will be illustrated for case study 1.

◇ **Step 6.***Model solution*
In order to solve the evaporator model represented by a set of PDAEs, the method of lines (included in ICAS-MoT) is used to obtain its discretized form. In this approach, ICAS-MoT does an automatic M-point discretisation applying centred finite difference. The resulting DAE system is solved using the Backward Difference Formula method (one of the numerical integration methods available within ICAS-MoT). In particular, for the short-path evaporator model, the discretisation is done for the coordinate "y" as shown in Figure 5.32. Good performance (i.e. the approximate solution converges to the true equation solution) can be achieved with a minimum value of $M = 10$ points, which gives a trade-off between the accuracy and a reasonable number of equations generated in the discretisation process, and because the numerical solutions of the temperature, velocity and concentration profiles are virtually unchanged as the number of discretisation points is increased.

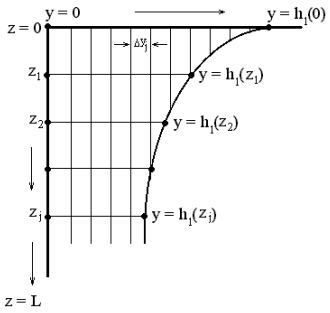


Figure 5.32: Discretisation scheme.

After discretisation we have $(12M + 2N + 5)$ total equations classified as: $(12M + N)$ ODEs and $(5 + N)$ AEs. There are $[17 + 13N + 12M)]$ total variables, which are classified as: $(6 + 2N)$ known, $(6 + 9N)$ model parameter, $[13M + N]$ dependent and $(5 + N)$ explicit variables. For the case of six components and $M = 10$ discretisation points we have 126 ODEs and 11 AEs; and 215 total variables, which are classified as: 18 known, 60 model parameter, 126 dependent and 11 explicit variables (note: DOF = 18 + 60 = 78).

◇ **Step 7.***Model Validation*
Sensitivity Analysis ICAS-MoT provides a sensitivity analysis option to determine the sensitivity of response variables (or model outputs) of the model due to changes in model parameters or input (design) variables. This is an important method for checking the quality of a given model (including the ro-

Table 5.20: Variable classification

Variable	**Notation**	**Units**	**Type of variable**	**Number of variables***
Constants				
Universal gas constant	R_g	$J(molK)^{-1}$	Known	1
Gravitational constant	g	ms^{-2}	Known	1
Pi	π	-	Known	1
Equipment dimensions				
Evaporator radius	R	m	parameter	1
Distance between evaporator - condenser		m	parameter	1
Evaporator length	L	m	parameter	1
Compound properties				
Molar weight	$M_{w,i}$	$Kgmol^{-1}$	parameter	N
Thermal conductivity	λ_i	$W(mK)^{-1}$	parameter	N
Density	ρ_i	Kgm^{-3}	parameter	N
Thermal capacity	Cp_i	$J(molK)^{-1}$	parameter	N
Heat of evaporation	ΔH_i^{vap}	$kJmol^{-1}$	parameter	N
Diffusion coefficient	D_i	m^2 s	parameter	N
Vapour pressure	P_i^{vap}	Pa	parameter	N
Liquid viscosity	η_i	Kg (m s)$^-$1	parameter	N
Activity coefficient	γ_i	-	parameter	N
Model parameters				
Number of intermolecular collisions	n	-	parameter	1
Mean path of vapour molecule	β	m	parameter	1
Surface ratio	F	-	Explicit	1
Anisotropy degree of the vapour	κ	-	Explicit	1
Pressure system effect	P/P_{ref}	-	parameter	1
Operating conditions				
Evaporator wall temperature	T_{w1}	K	Known	1
System Pressure	P	Pa	Known	1
Feed Temperature	T_F	K	Known	1
Feed composition of each compound	$C_{i,0}$	mol m$^-$3	Known	N
Feed Flow of each compound	$I_{i,0}$	mol s$^-$1	Known	N
*Outlet variables***				
Velocity profile	$v(y,\ z)$	m s$^-$1	Dependent	1
Temperature profile	$T(y,\ z)$	K	Dependent	1
Composition profile of each comp.	$C_i(y,z)$	mol m$^-$3	Dependent	N
Total composition	$C(z)$	mol m$^-$3	Explicit	1
Effective evaporation rate of each comp.	$k_i(z)$	mol m$^-$2s$^-$1	Explicit	N
Exit flows of each compound	$I_i(z)$	mol s$^-$1	Dependent	N
Total exit flow compound	$I(z)$	mol s$^-$1	Explicit	1
Thickness film profile	$h_1(z)$	m	Explicit	1

* N = Number of compounds
** y = coordinate and z = coordinate
coordinate (independent variable)

bustness and reliability) as well as identifying the most important (sensitive) design variables. ICAS-MoT performs this analysis by means of systematic perturbations that involves changing the value of one or more selected variables and calculating the resulting change in the unknown (output) variables. Changes in parameter variables can be assessed one at a time to identify the responses of key (output) variables. If a small change in a parameter results in relatively large change in the response variable, the response variable is said to be sensitive to that parameter. This may mean that the parameter has to be determined very accurately or that the process/operation has to be redesigned for lower sensitivity. For the short-path evaporator model, the performance of sensitivity analysis is important to identify mainly the design variables or operation conditions, so that the yield and purity of the desired chemical product can be improved.

Case study 1: Purification of a reaction mixture

Consider a mixture containing glycerol and caprylic esters in the form of caprylic mono-, di-, and triglycerides. The objective is to use a short path evaporator to remove the glycerol (as distillate) from the ester mixture. All the values of the input variables are given in Tables 5.21, 5.22 and 5.23 (where the superscripts 1, 2, 3, 4 denote the compounds glycerol, mo-, di-, and triglyceride, respectively).

The equations 5.139-5.144 for temperature dependent properties calculation were used (Table 5.21.)

$$P_i^{vap} = exp[A_i + B_i/T + C_i * ln(T) + D_i * T^{E_i}] \tag{5.139}$$

$$\Delta H_i^{vap} = A_i * (1 - T/Tc_i)^{[}B_i + C_i * T/Tc_i + D_i * (T/Tc_i)^2] \tag{5.140}$$

$$Cp_i = A_i + B_i * T + C_i * T^2 + D_i * T^3 + E_i * T^4 \tag{5.141}$$

$$\rho_i = A_i/B_i^{[1+(1-T/C_i)^{D_i}]} \tag{5.142}$$

$$\eta_i = exp[A_i + B_i/T + C_i * ln(T) + D_i * T^{E_i}] \tag{5.143}$$

$$\lambda_i = A_i + B_i * T + C_i * T^2 + D_i * T^3 + E_i * T^4 \tag{5.144}$$

Model Results: Case 1

The solution of the evaporator model provides simulated values of the outlet flows (of each compound) and compositions in both distillate and residue, as highlighted in Figures 5.33 and 5.34. The simulated outlet compound flows and

Table 5.21: Compound properties - temperature dependent equations*

Property	Eq. No.
Vapour Pressure $[Pa]$	5.139
Heat of vaporization $[J/kmol]$	5.140
Liquid Heat Capacity $[\mathrm{J}/(kmol * K)]$	5.141
Density $[kmol/m^3]$	5.142
Liquid Viscosity $[Kg/ms]$	5.143
Thermal Conductivity $[W/(mK)]$	5.144

*: CAPEC Data base, (Gani 2002).

Table 5.22: Compound properties - temperature dependent constant values.*

$Eq.No.$	A_i	B_i	C_i	D_i	E_i
(5.139)	-254.000^1	50.600^1	43.000^1	$-3.99e{-}5^1$	0
	168.935^2	-19323.687^2	-20.944^2	$4.89e{-}6^2$	0
	155.638^3	-18923.011^3	-19.122^3	$4.13e{-}6^3$	0
	159.805^4	-20239.392^4	-19.610^4	$3.90e{-}6^4$	0
(5.140)	215.000^1	2.120^1	-2.76^1	1.260^1	0
	179.739^2	1.9992^2	-2.605^2	1.206^2	0
	184.886^3	2.101^3	-2.749^3	1.241^3	0
	146.747^4	1.150^4	-1.420^4	0.798^4	0
(5.141)	3080.000^1	-23.900^1	0.0786^1	$-1.15e{-}4^1$	$6.31e{-}8^1$
	1443.389^2	-7.049^2	0.0268^2	$-2.57e{-}5^2$	$1.20e{-}8^1$
	655.296^3	-0.469^3	0.0033^3	$-3.96e{-}6^3$	$1.59e{-}9^1$
	401.8804^4	1.2511^4	0.00087^4	$-1.61e{-}6^4$	$6.00e{-}10^1$
(5.142)	0.947^1	0.2498^1	723.0^1	0.152^1	0
	0.687^2	0.276^2	780.0^2	0.413^2	0
	0.582^3	0.356^3	800.0^3	0.526^3	0
	0.211^4	0.256^4	883.0^4	0.365^4	0
(5.143)	-237.03^1	16739.0^1	31.734^1	0.0^1	0
	-12.112^2	5212.908^2	$-3.52e{-}5^2$	$2.12e{-}11^2$	0
	-10.478^3	4636.307^3	$-6.39e{-}5^3$	$3.30e{-}11^3$	0
	-8.988^4	4059.733^4	$-1.88e{-}5^4$	$-9.82e{-}12^4$	0
(5.144)	0.2508^1	0.000113^1	0.0^1	0.0^1	0
	0.0186^2	0.001067^2	$-3.60e{-}6^2$	$4.71e{-}9^2$	0
	0.07372^3	0.000404^3	$-1.47e{-}6^3$	$1.85e{-}9^3$	0
	-0.07154^4	0.001262^4	$-3.45e{-}6^4$	$-3.87e{-}9^4$	0

*: CAPEC Data base, (Gani 2002).

Table 5.23: Problem data

Variable	**Notation**	**Value**	**Units**
Constants			
Universal gas constant	R_g	8.31451	J (mol K)$^{-1}$
Gravitational constant	g	9.81	m s^{-2}
Pi	π	3.1416	-
Equipment dimensions			
Evaporator radius	R	0.36	m
Distance between evaporator - condenser	d	0.065	m
Evaporator length	L	3	m
Compound properties			
Molar weight	$M_{w,i}$	92.095[1]	Kg mol^{-1}
		218.29[2]	
		344.48[3]	
		470.68[4]	
Activity coefficient	γ_i	0.9872[1]	-
		0.9547[2]	
		0.9967[3]	
		0.9874[4]	
Model parameters			
Number of intermolecular collisions	n	5	-
Mean path of vapour molecule	β	0.065	m
Pressure system effect	P_{ref}	101325	Pa
Operating conditions			
Evaporator wall temperature	T_{w1}	473	K
System Pressure	P	10	Pa
Feed Temperature	T_F	353	K
Feed composition of each compound	$C_{i,0}$	3.55596[1]	Kmol m^{-3}
		0.9708[2]	
		0.1618[3]	
		0.3236[4]	
Feed Flow of each compound	$I_{i,0}$	0.055[1]	mol s^{-1}
		0.015[2]	
		0.025[3]	
		0.005[4]	

compositions in the distillate are, (0.05210, 0.00392, 0.00145, $5.0x10^{-5}$) mol s^{-1}, and (90.57, 6.82, 2.52, 0.09) % mol, respectively; while in the residue are, (0.00294, 0.01108, 0.02355, 0.00495) $mols^{-1}$, and (6.91, 26.06, 55.39, 11.64) % mol, respectively. According to these results there is a good separation of glycerol in the distillate, and there remains just a little of glycerol in the residue. Moreover, it can be seen in Figure 5.33 and 5.34 that at the top of the evaporator ($z < 0.5\,m$) the flow rates are almost constant, meaning a very low evaporation rate that produces a thick film in the evaporating surface (as shown in Figure 5.35) and almost no formation of a condensing film. This explains the behaviour of the composition profile between $z = 0$ to $z = 0.5$ m, where practically before $z = 0.25\ m$ there is no film and between $z = 0.25$ to $z = 0.5\ m$ the condensing film starts growing, mainly with glycerol product.

Figure 5.35 shows the temperature, velocity and thickness profiles in the evaporating surface. It can be seen that at the end of the evaporator, the temperature and film thickness is almost constant, while the velocity is still decreasing slowly.

To illustrate the model reformulation option in ICAS-MoT, two design problems are formulated and solved. In *Design-1*, the equipment dimensions (evaporator radius, gap and length) are determined to produce a specific separation (i.e., the outlet flows on the residue stream), while in *Design-2*, the optimal operation conditions (i.e., feed temperature and evaporator pressure) are determined knowing the exit concentrations in the evaporating surface. Both design problems can be formulated in two ways as: (a) An inverse problem where the output (target) variables, which are now known, are interchanged with the corresponding input variables, which are now unknown, while keeping the DOF unchanged in the model equations. That is, the same set of model equations are solved but for a different set of known and unknown variables. (b) A mathematical programming (optimisation) problem to minimize the quadratic deviation of the desired outputs from the target, subject to a set of design variables (the equipment dimensions). In this case study, the second option, solving an optimisation problem is highlighted. The two optimal design problems are formulated as: Find optimal values for

$$\mathbf{u} = [R, d, L] \qquad \textit{design-1} \tag{5.145}$$

or

$$\mathbf{u} = [T_F, P] \qquad \textit{design-2} \tag{5.146}$$

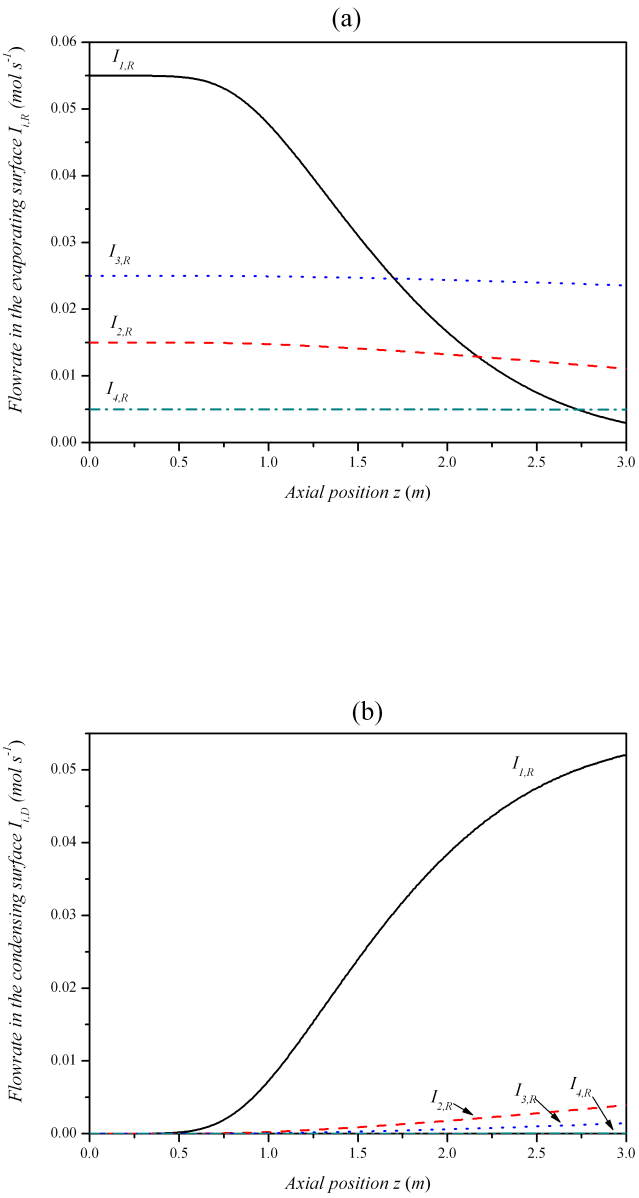


Figure 5.33: Flow profile in: (a) residue (I_R) and (b) the distillate (I_D).

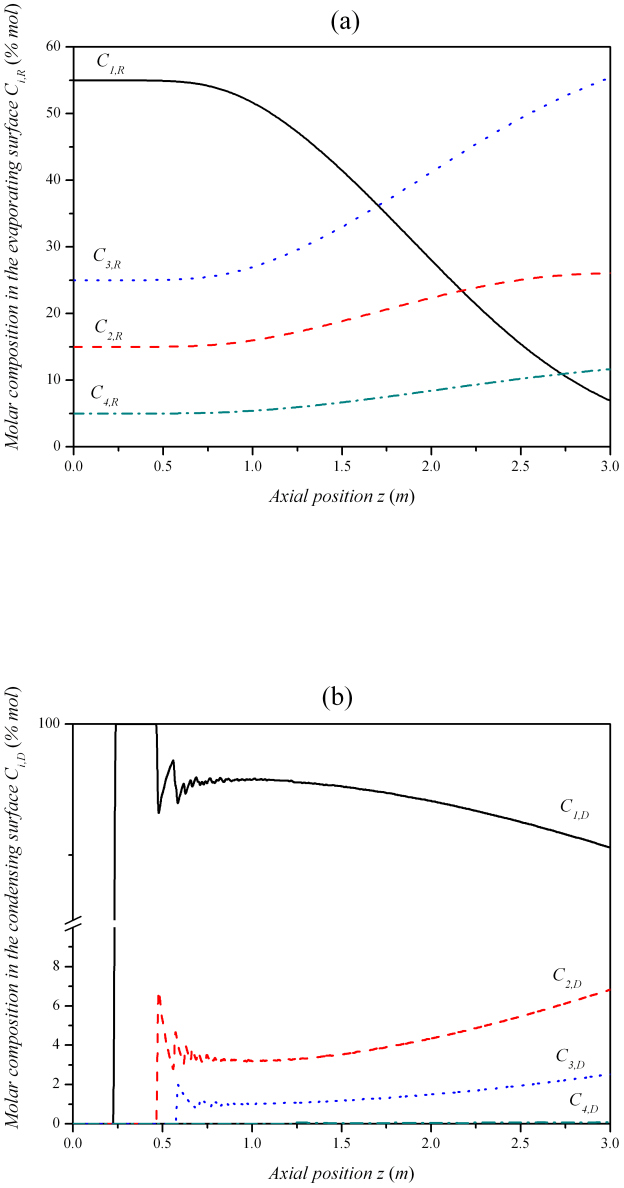


Figure 5.34: Composition profile in: (a) the residue ($C_{i,R}$) and (b) the distillate ($C_{i,D}$).

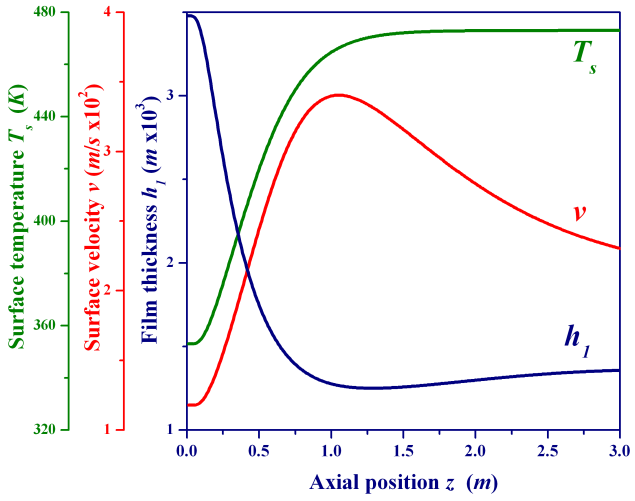


Figure 5.35: Temperature, velocity and film thickness profiles in the evaporating surface.

by minimizing the objective function:

$$\min_{u} \; J = \sum_{i=1}^{n} \left(\frac{y_i - \bar{y}_i}{\bar{y}_i} \right)^2 \tag{5.147}$$

where

$$\mathbf{y} = [I_{1,R}, I_{2,R}, I_{3,R}, I_{4,R}] \qquad \textit{design-1} \tag{5.148}$$

or

$$\mathbf{y} = [C_{1,R}, C_{2,R}, C_{3,R}, C_{4,R}] \qquad \textit{design-2} \tag{5.149}$$

Subject to the model equations 5.125- 5.143, and with bounds $\mathbf{u}_{\min} \leq \mathbf{u} \leq \mathbf{u}_{\max}$.

Results of the optimisation problems are presented in Table 5.24, where the optimal input values were obtained by reaching the defined targets. These two examples highlight the usefulness and application of the generalized short-path evaporator model. In a similar way, problems where the objective function would represent the minimisation of total operating cost (or maximisation of product throughput) under certain product quality specifications (e.g. lower bound in the purity and/or product flowrates) and other operating and design limitations can be solved. In such cases the decision variables could be both the design parameters (e.g. size of the equipment) and operating conditions (e.g. pressure, temperature, etc).

Case study 2: A pharmaceutical mixture

This case study involves an industrial pilot plant process for the production of a drug where impurities are removed from a mixture that leaves the reaction stage. The compounds representing the active pharmaceutical ingredient (API) are formed in the reaction stage and the resulting liquid mixture is composed of six heat sensitive compounds (that will be called A, B, C, D, E and F for reasons of confidentiality). Compound A is the lightest and most volatile compound while compound F is the heaviest having the highest boiling point. The role of the short-path evaporator is to separate the API (consisting of mainly C, D and E) together with the inert component F (as the residue product) from the multi-component feed mixture. Data from the pilot plant for the feed and outlet compound flows (distillate and residue) are available (Table 5.25). However, as the pilot plant data is restricted by confidentiality agreement, details of the data for the components in this case study (compound properties, evaporator design, operation conditions, etc.) are not reported here. Instead, the simulation results obtained from the model are compared with the pilot plant data to validate the evaporator model (see Table 5.25).

Model Result: Case 2

The feed flows as well as experimental and calculated flows of the distillate and

Table 5.24: Redesign of the short-path evaporator. (a) Example 1: Optimal evaporator dimensions, (b) Example 2: Optimal operation conditions.

Variable	*Example 1*		*Units*		*Variable*	*Example 2*		*Units*
Outputs:	*Desired*	*Reached*			*Outputs:*	*Desired*	*Reached*	
$I_{1,R}$	0	$3x10^{-6}$	$mols^{-1}$		$C_{1,R}$	1	0.0084	$\%mol$
$I_{2,R}$	0.0055	0.00549	$mols^{-1}$		$C_{2,R}$	19	18.8501	$\%mol$
$I_{3,R}$	0.02	0.0205	$mols^{-1}$		$C_{3,R}$	65	65.7868	$\%mol$
$I_{4,R}$	0.005	0.00484	$mols^{-1}$		$C_{4,R}$	15	15.3547	$\%mol$
Optimal inputs:					*Optimal inputs:*			
R	0.1282		m		T_F	360.75		K
L	3.9321		m		P	15		Pa
d	0.0568		m		T_{w1}	490.01		K

residue are reported in Table 5.25, where it can be seen that the model is able to predict the outlet flow rates in a satisfying way. This is purely prediction, as none of the pure compound property model parameters in Eqs. (5.139)-(5.144) were adjusted to match the pilot plant data. The chemical product (C, D, E and F) is obtained as the residue in the short-path evaporator, and the surface velocity, temperature, thickness and some flow rates are shown for the evaporating film in Figures 5.36-5.38. The rise in surface temperature (Figure 5.36) is mainly related to the increase in the evaporation rate of compounds A and B. In fact, A and B are the main compounds that are evaporated from the mixture and are obtained as distillate product (see Table 5.25 and Figure 5.37). The surface velocity (Figure 5.36) shows a rapid increase in the first part of the evaporator axial position achieving a maximum point, and then decreasing slightly due to the decrease of the total evaporation rate. Figure 5.36 also shows the dependence of the film thickness throughout the evaporator cylinder axis. Film thickness decreases with the increasing surface temperature due to evaporation. Both film surface temperature and film thickness turn asymptotic as soon as a constant film thickness has formed. Moreover, the evaporator model allows prediction of the entire temperature profile along both y and z positions, as it is shown in Figure 5.38. This figure is important because shows that the temperature has a continuous variation, meaning that the effective rate of evaporation is highly affected [see boundary condition defined by Eq. (5.132)] by the temperature modelling, and therefore the flow rates [Eq. (5.127)] of distillate and residue are also highly dependent on the modelling of the temperature profile.

The feed temperature T_f is one of the important operational parameters that determines the rate of evaporation and yield of the product in the short evaporator operation, which in turn are determined mainly by the film surface temperature T_s and film thickness h_1. Figure 5.39 shows the behaviour of the film surface temperature as a function of the axial distance (evaporator length) at five different feed temperatures. It can be seen that all the feed temperatures achieve the same asymptotic film surface temperature. Figure 5.39 shows the position dependence of the film thickness for the same five feed temperatures. It can be seen that all the film thicknesses achieve the same asymptotic value at the end of the evaporator. However, as the feed temperature increases, the

Table 5.25: Pilot plant data and calculated flow rates of distillate and residue.

Compound	Pilot plant flows (mol s^{-1})			Calculated flows ($mols^{-1}$)	
	Feed I_0	Residue I_R	Distillate I_D	Residue I_R	Distillate I_D
A	1.70E-05	0.00E+00	1.70E-05	0.00E+00	1.70E-05
B	3.39E-06	0.00E+00	3.39E-06	7.59E-07	2.63E-06
C	1.31E-02	1.24E-02	7.25E-04	1.29E-02	1.71E-04
D	5.28E-05	5.22E-05	5.81E-07	5.27E-05	8.36E-08
E	6.03E-04	6.00E-04	1.89E-06	6.01E-04	2.11E-06
F	1.85E-04	1.85E-04	0.00E+00	1.85E-04	0.00E+00

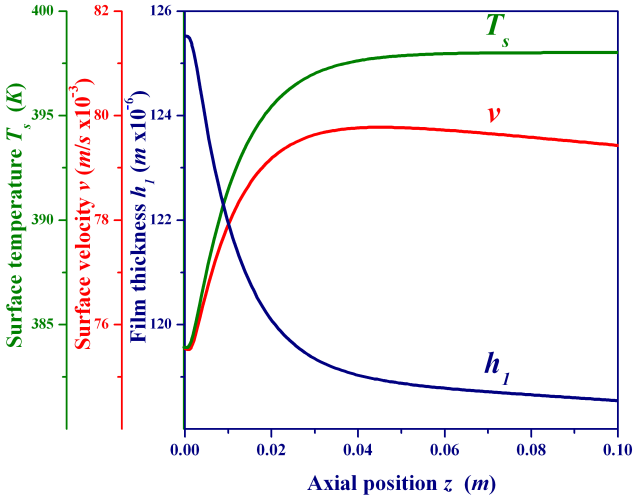


Figure 5.36: Surface temperature, surface velocity and thickness in the evaporating film.

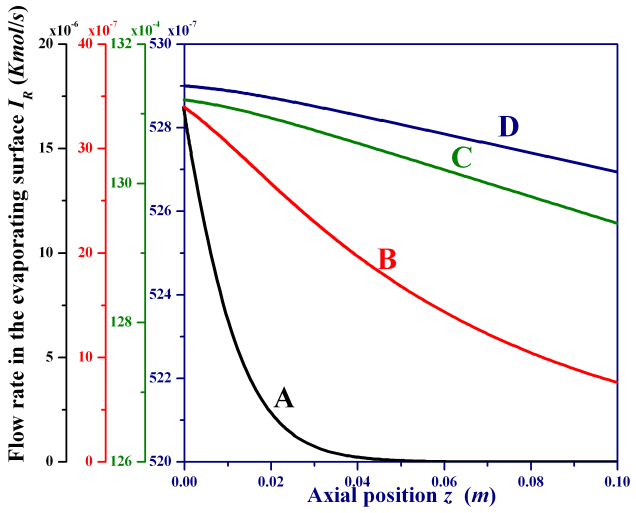


Figure 5.37: Flow rate for compounds A, B, C, D.

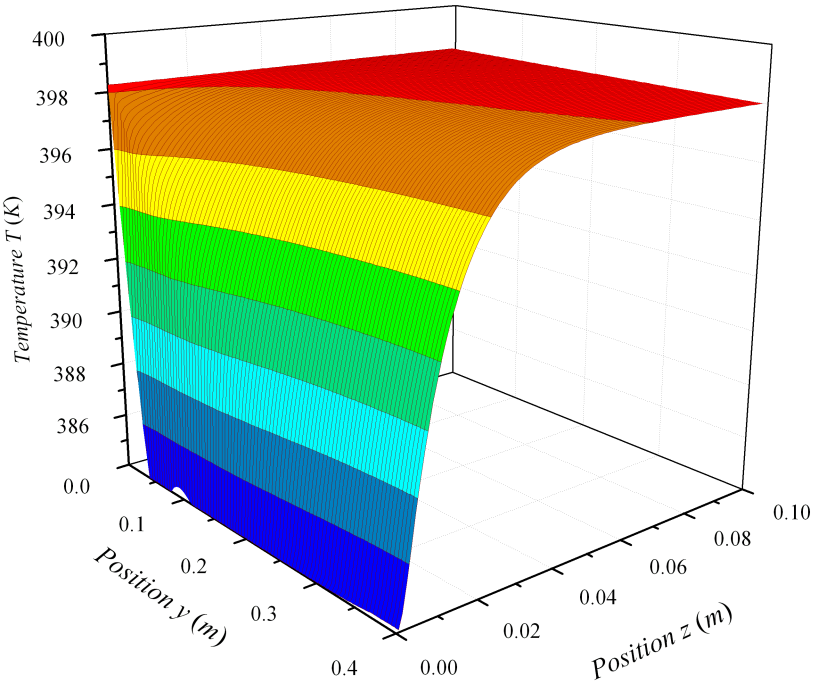


Figure 5.38: Temperature profile as a function of axial (z) and (y) positions.

film thickness decreases, meaning that at higher feed temperatures, a higher evaporation rate and a better product yield are obtained.

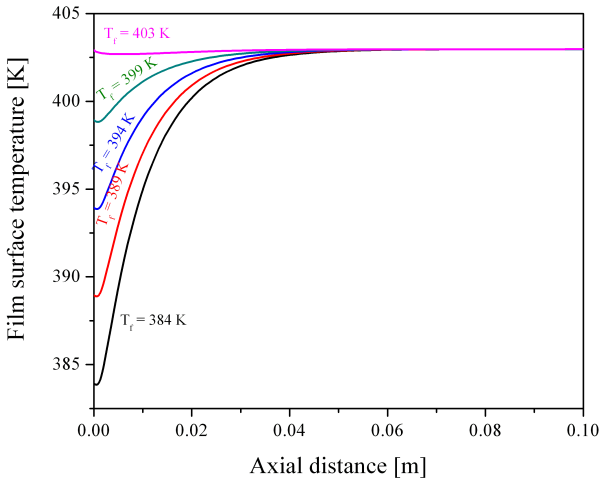


Figure 5.39: Effect of the feed temperature on the film surface temperature at $T_w = 402$ K.

Finally, a sensitivity analysis was performed through the corresponding option in ICAS-MoT. The evaporator diameter, the distance between the evaporator and condenser (gap), and the operation pressure were selected for analysis with respect to their effect on the film surface temperature and product flow rates. The results are highlighted in Figures 5.41-5.2.2, where it can be seen that the evaporation process is not very sensitive to changes in the evaporation gap, while it is quite sensitive to changes in the evaporator diameter and the operation pressure. Moreover, the film surface temperature and flow rate of component F are not as sensitive to the changes in the selected parameters as the flow rate of components B, C and D. In fact, the most sensitive variable is the flow rate of component C, which also happens to be the component, which is present in the largest amount in the feed mixture. The sensitivity analysis suggests that the main design variables for this particular process are the operating pressure and the evaporator diameter.

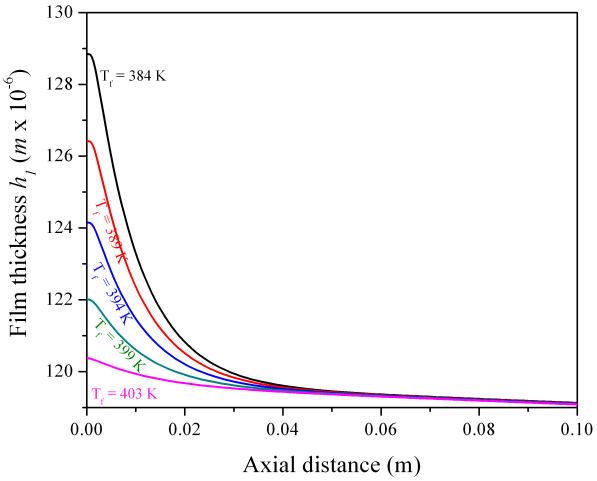


Figure 5.40: Effect of the feed temperature on the film thickness at $T_w = 402$ K.

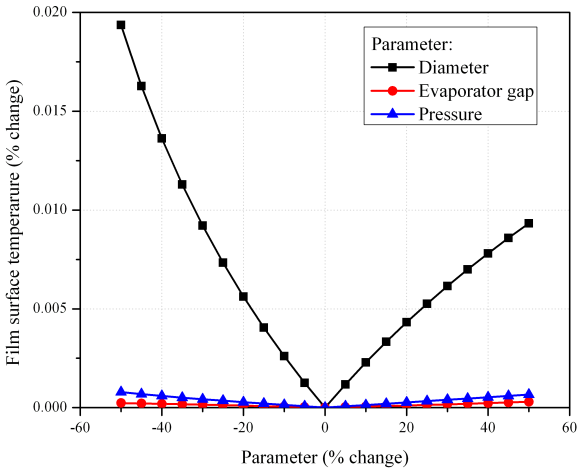


Figure 5.41: Sensitivity on the film surface temperature.

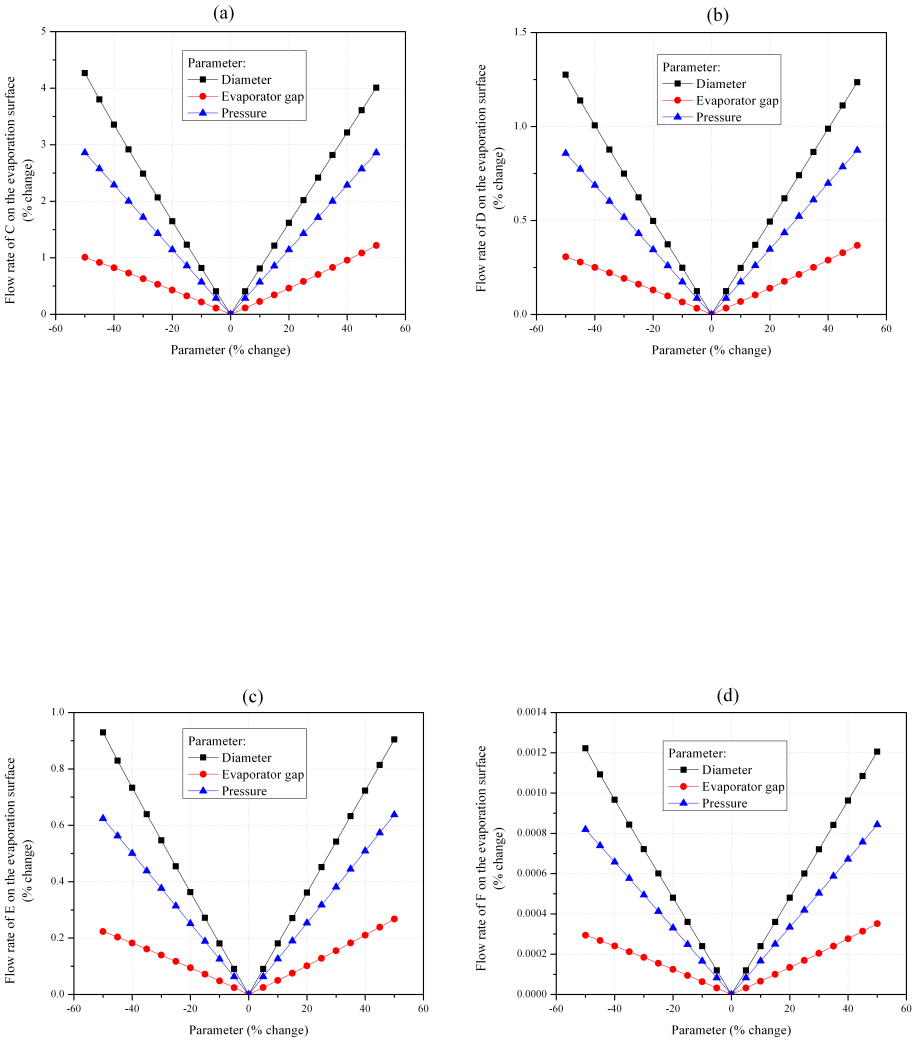


Figure 5.42: Sensitivity analysis on products flow rates for components C, D, E and F; figures (a), (b), (c) and (d) respectively.

Case summary

A generalized short-path evaporator model has been presented and its application for the purification of chemical products from multi-component mixtures has been highlighted. ICAS-MoT has been employed to perform the various modelling tasks, thereby, providing a means for rapid and reliable study of short-path evaporation processes. The applicability of the developed model as well as the various options of the modelling framework have been illustrated through two case studies, one of which is taken from an industrial process. A sensitivity analysis has been performed to identify the most important model parameters with respect to design and model identification (parameters that may be adjusted to match the available plant data). For an industrial pilot plant, model identification was not necessary as very good match was obtained without any adjustment of parameters. Finally, the results highlight the importance of a general-purpose and easy to use modelling toolbox for computer-aided design and analysis of complex process operations.

5.2.3 Dynamic system (Polymerisation of MMA)

The production of synthetic polymers (i.e. commodity plastics, engineering plastics and speciality polymers) is one of the most important worldwide industries. Each year more than 100 million tons of synthetic polymers are produced worldwide (Ray and Villa 2000). Many synthetic polymers are manufactured through either mass, solution, emulsion and suspension polymerisations. From an industrial perspective, the main operations during the polymerisation processes are to increase or optimise the productivity; to maintain the polymer grade (i.e. composition, conversion and molecular weight) or the flexibility in obtaining different polymer grades; to reduce the process time and therefore the cost of production and energy consumption; to reduce waste emissions; while maintaining acceptable ranges of operability (safety and stability). Therefore, the knowledge of the instantaneous polymer characteristics (conversion, composition, molecular weight, solid fraction, etc.) during the production process is important for enhancing industrial competitiveness (through efficient operation and control, flexible design and optimal cost). To achieve this, several challenges need to be overcome since the polymerisation processes are complex due to their highly non-linear dynamic behaviour, the existence of steady state multiplicity, the tendency to unstable motions, the strong input-output coupling and the extreme sensitivity of the steady states to changes in the operational (design) parameters.

◇ **Step 1.** *System description*
Some common operational difficulties encountered in polymerisation reactors are the infrequently available measurements of the product grade, the change of the production requirements and the presence of disturbances, the fault occurrences of sensors/process components/actuators and abnormal process behaviour. The solution of these problems through model-based analysis requires a good and thorough understanding of the process behaviour that will enable an appropriate process model that can describe the steady state and dynamic behaviour and through which design operational strategies can be developed.

Figure 5.43 shows the procedure that should be followed to implement and solve the problem described above using ICAS-MoT. This problem deals with a DAE system.

The work-flow that should be followed to solve the problem is highlighted with dark grey colour, whereas the blocks in light grey means that the path is inactivate for problems such as the one tackled here. This work-flow involves the main steps to set up the DAE model and an algebraic solver.

◇ **Step 2.** *Problem definition*
This example involves the analysis of the open-loop (i.e. without control) and closed-loop (i.e. including a control) behaviour of a continuous stirred tank reactor (CSTR), where the methyl methacrylate (MMA) polymerisation reaction takes place (Figure 5.44).

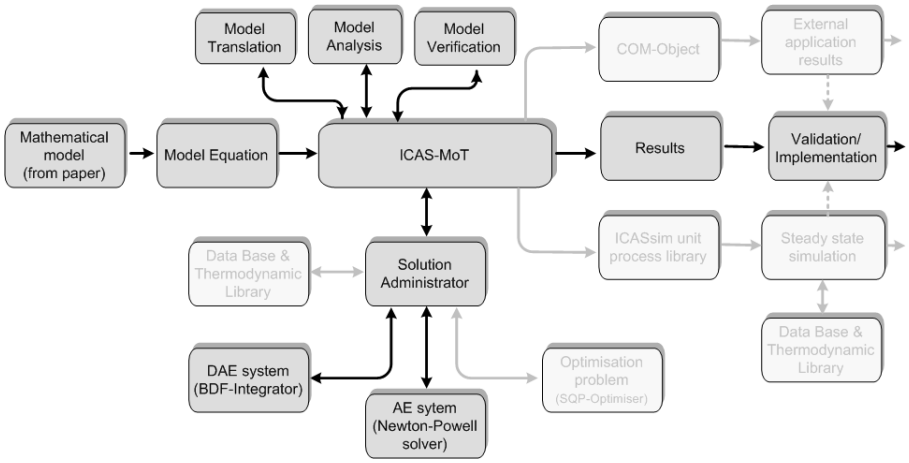


Figure 5.43: Work-Flow and tools used from ICAS-MoT for steady state and dynamic simulation (case study 3)

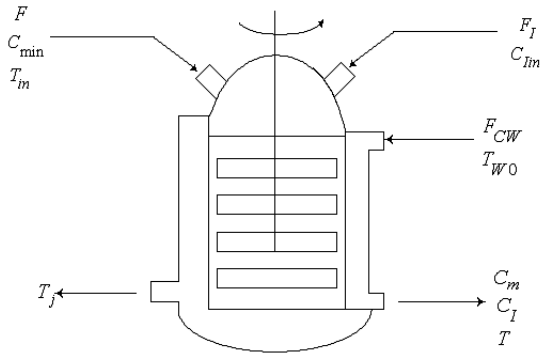


Figure 5.44: Polymerisation reactor flowsheet.

The process is described as the bulk free-radical polymerisation of MMA using AIBN (azo-bis-iso-butyronitrile) as initiator and toluene as solvent. The reaction is exothermic and a cooling jacket is used to remove the heat of reaction (Silva et al., 2001).

Controlling factors. The reaction mechanism of free-radical MMA polymerisation consist of the following steps:

Initiation:

$$I \xrightarrow{k_0} 2R \tag{5.150}$$

$$R + M \xrightarrow{k_I} P_1 \tag{5.151}$$

Propagation:

$$P_i + M \xrightarrow{k_p} P_{i+1} \tag{5.152}$$

Monomer transfer:

$$P_i + M \xrightarrow{k_{fm}} P_i + D_i \tag{5.153}$$

Disproportionation termination:

$$P_i + P_j \xrightarrow{k_{td}} D_i + D_j \tag{5.154}$$

Where I, P, M, R and D stand for initiator, polymer, monomer, radicals, and dead polymer, respectively.

◇ **Step 3.** *Model construction*

Model assumptions. The assumptions to develop the mathematical model are:

$\mathcal{A}_1$. contents of the reactor are perfectly mixed,
$\mathcal{A}_2$. constant density and heat capacity of the reaction mixture,
$\mathcal{A}_3$. density and heat capacity of the cooling fluid stay constant,
$\mathcal{A}_4$. uniform cooling fluid temperature,
$\mathcal{A}_5$. the reactions only happen inside the reactor,
$\mathcal{A}_6$. there is no gel effect (the conversion of monomer is low and the proportion of solvent in the reaction mixture is very high),
$\mathcal{A}_7$. constant volume of the reactor,
$\mathcal{A}_8$. the polymerisation reactions occur by the free-radical mechanism, and
$\mathcal{A}_9$. only the propagation reactions generate a reaction heat [i.e. $(-\Delta H_p) \neq 0$ and ΔH_0, ΔH_I, ΔH_T, ΔH_{tc}, $\Delta H_{td} = 0$].

When both the previous assumptions and the identified controlling factors are taken into account, the mathematical model of the MMA polymerisation takes the following form:

$$\frac{dC_m}{dt} = \frac{F\left(C_{\min} - C_m\right)}{V} - \left(r_p + r_T\right) := f_m, \qquad C_m\left(0\right) = C_{m,0} \tag{5.155}$$

$$\frac{dC_I}{dt} = \frac{\left(F_I C_{I_{in}} - F C_I\right)}{V} - r_I := f_I, \qquad C_I\left(0\right) = C_{I,0} \tag{5.156}$$

$$\frac{dT}{dt} = \frac{F\left(T_{in} - T\right)}{V} + \frac{\left(-\Delta H_p\right) r_p}{\rho C_P} - \frac{UA}{\rho C_P V}\left(T - T_j\right) := f_T, \qquad T\left(0\right) = T_0 \tag{5.157}$$

$$\frac{dT_j}{dt} = \frac{F_{CW}\left(T_{W0} - T_j\right)}{V_0} + \frac{UA}{\rho_W C_{PW} V_0}\left(T - T_j\right) := f_{Tj}, \qquad T_j\left(0\right) = T_{j,0} \tag{5.158}$$

where:

$$P_0 = \sqrt{\frac{2f^* k_I C_I}{k_{td} + k_{tc}}} \tag{5.159}$$

$$k_r = A_r e^{-E_r/RT}, \qquad r = p, fm, I, td, tc \tag{5.160}$$

$$r_p = k_p C_m P_0 \tag{5.161}$$

$$r_T = k_{fm} C_m P_0 \tag{5.162}$$

$$r_I = -k_I C_I \tag{5.163}$$

The mathematical model [equations 5.155- 5.163] correspond to a system of Differental-Algebraic Equations (DAEs), which can be written in the following compact notation:

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}\left(\mathbf{x}, \mathbf{u}, \mathbf{d}, t\right) \tag{5.164}$$

where $\mathbf{x}$ is the vector of states, $\mathbf{u}$ is the vector of process inputs, $\mathbf{d}$ is the vector of exogenous inputs (i.e. system disturbances) and $\mathbf{f}$ is the vector of dynamic functionalities:

$$\mathbf{x} = \begin{bmatrix} C_m \\ C_I \\ T \\ T_j \end{bmatrix}, \mathbf{u} = \begin{bmatrix} F_I \\ F_{CW} \end{bmatrix}, \mathbf{d} = \begin{bmatrix} C_{m,in} \\ C_{I,in} \\ T_{in} \\ T_{w0} \end{bmatrix}, \mathbf{f} = \begin{bmatrix} f_m \\ f_I \\ f_T \\ f_T j \end{bmatrix} \tag{5.165}$$

Figure 5.45 shows a block-diagram of the open-loop process, showing the dependence of the variables. This diagram will be used afterwards to design a controller and model the closed-loop process.

◇ **Step 3.** *Model Analysis*

Problem A: Nonlinear Open-Loop Analysis

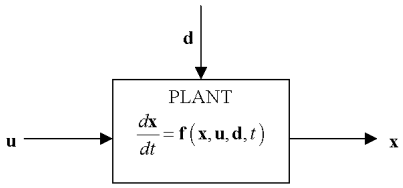


Figure 5.45: Open loop plant

Steady state(Model Analysis)
In order to solve the reactor model in steady state, equations 5.155- 5.163 must be equal to zero. The corresponding ICAS-MoT steady-state model is given in appendix D. The variable and equation classification of this model is as follows:

(a) it has a total of 13 algebraic equations, sorted as 9 explicit and 4 implicit equations; and

(b) there are a total of 41 variables, sorted as 27 known, 1 parameter (constant), 9 explicit, and 4 unknown.

Therefore the degree of freedom (DOF = number of variables - number of equations) is 28, which means that 28 variables must be specified to solve the problem. These variables are the 27 known ones given in Tables 5.26 - 5.28 and the only parameter the universal gas constant, R.

◇ **Step 5.***Model data needed*
All kinetic and physicochemical parameters required to solve the polymerisation model [equations 5.155- 5.163] are reported elsewhere (Brandrup and Immergut, 1975). As a reference case, the reactor design and operating conditions were taken from Silva et al. (2001). All these data are summarize in the Tables 5.26 - 5.28.

◇ **Step 6.***Model solution*
Steady state solution.
Considering initial states given in Table 5.29, the steady-state model was solved by using the Newton-Powell method. The results reported in Table 5.30 were obtained, where it can be seen that tree steady states were found.

◇ **Step 7/8.** *Model verification/validation*
To define the stability of the three steady states, an analysis of the Jacobian of the nonlinear system was done. The Jacobian matrix was evaluated at the three different steady states, then the eigenvalues of these matrices were found as reported in Table 5.31. It can be seen the steady states 1 and 3 are stable

Table 5.26: Kinetic Parameters: MMA polymerisation

Parameter	Value	Units
E_p	1.8283 x 10^4	$kJ/kgmol$
E_I	1.2877 x 10^5	$kJ/kgmol$
E_{fm}	7.4478 x 10^4	$kJ/kgmol$
E_{tc}	2.9442 x 10^3	$kJ/kgmol$
E_{td}	2.9442 x 10^3	$kJ/kgmol$
A_p	1.77 x 10^9	$m3/(kgmolh)$
A_I	3.792 x 10^{18}	$1/h$
A_{fm}	1.0067 x 10^{15}	$m^3/(kgmolh)$
A_{tc}	3.8283 x 10^{10}	$m^3/(kgmolh)$
A_{td}	3.1457 x 10^{11}	$m^3/(kgmolh)$
f^*	0.58	-
ΔH_p	57800	$kJ/kgmol$

Table 5.27: Physico-chemical Properties for the MMA problem

Parameter	Value	Units
r	866	kg/m^3
r_W	1000	kg/m^3
Cp	2	$kJ/(kgK)$
Cp_W	4.2	$kJ/(kgK)$

Table 5.28: Reactor design and operation conditions

Parameter	Value	Units
U	720	$kJ/(hkm^2)$
A	2	m^2
V	0.1	m^3
V_0	0.02	m^3
F	1	m^3/h
F_I	0.0032	m^3/h
F_{CW}	0.1588	m^3/h
C_{min}	6.4678	$kgmol/m^3$
C_{I_in}	8	$kgmol/m^3$
T_{in}	350	K
T_{W0}	293.2	K

Table 5.29: Initial state values

	State 1	State 2	State 3	Units
$C_{m,0}$	0	6	2	$kgmol/m^3$
C_I	0.5	0	0	$kgmol/m^3$
T	300	365.5	500	K
T_j	300	335	390	K

Table 5.30: Steady-state solution

	State 1	State 2	State 3	Units
C_m	5.9658	5.88881	2.36379	$kgmol/m^3$
C_I	0.02492	0.02473	0.000177	$kgmol/m^3$
T	351.394	353.419	436.197	K
T_j	332.973	334.357	390.931	K

(because all real parts Re_{i1}, Re_{i2}, Re_{i3} and Re_{i4} are negative), while the steady state 2 corresponds to an unstable one (because Re_{24} is positive).

Table 5.31: Eigenvalues of the MMA polymerisation process

	Re_{i1}	Im_{i1}	Re_{i2}	Im_{i2}	Re_{i3}	Im_{i3}	Re_{i4}	Im_{i4}	
Steady state 1	-30.533	0	-10.064	0.125	-10.064	-0.125	-0.632	0	S
Steady state 2	-30.185	0	-10.072	0.146	-10.072	-0.146	0.663	0	U
Steady state 3	-1398.2	0	-21.14	4.575	-21.14	-4.575	-15.778	0	S

Steady state: Multiplicity analysis. In order to find the region of the steady states multiplicity, bifurcation diagrams were built using two continuation parameters: $\alpha = F_I$ (i.e. the feed flow of initiator) and $\beta = T_{w0}$ (i.e. cooling fluid feed stream temperature). The bifurcation diagrams were obtained using the CONT (continuation) method (also included in ICAS-MoT). The modified ICAS-MoT model is given in appendix D. The results are shown in Figure 5.46, where it can be seen that there are two stable branches (solid line), a middle unstable branch (dashed line), and the corresponding reference steady-states are marked with small circles. Here it is important to mention that the desired operating point is the middle unstable steady state, which is difficult to control due to its closeness to the bifurcation point.

$\diamond$*Model Modification*
The model above is now used in its dynamic form. This will highlight that the same ICAS-MoT code with a small modifications can be used in different model configurations.

Dynamic behaviour: Model analysis. In order to solve the reactor model dynamically, the DAE system [equations 5.155- 5.163] must be solved simultaneously. The corresponding ICAS-MoT dynamic model is given in appendix D. The variable and equation classification of this model, obtained through ICAS-MoT is as follows:

(a) it has a total of 13 equations, sorted as 9 algebraic and 4 differential equations; and

(b) there are a total of 41 variables, sorted as 27 known, 1 parameter (constant), 9 explicit, and 4 dependent.

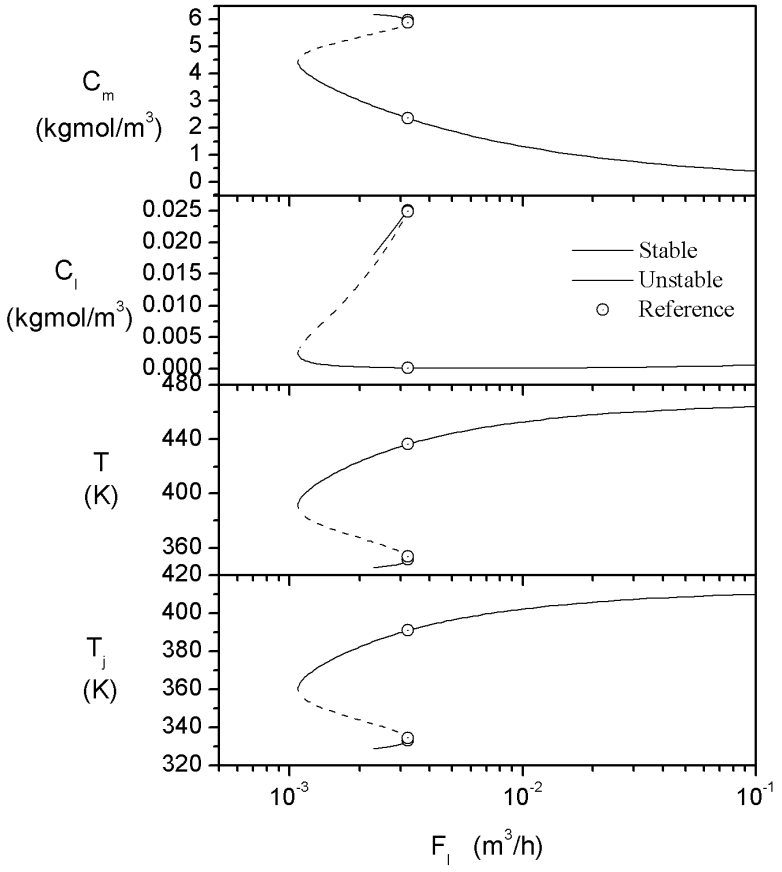


Figure 5.46: Multiplicity of steady states: F_I as bifurcation parameter

Therefore, the degree of freedom (DOF = number of variables - number of equations) is 28, which means that 28 variables must be specified to solve the problem. These variables are the 27 known ones given in Tables 5.26 - 5.28 and the parameter that corresponds to the universal gas constant, R.

Dynamic behaviour: Model solution. The dynamic simulations were set-up through ICAS-MoT by selecting the BDF method for integration of the DAE system. Three cases were considered.

1. Case I. Two initial states were considered: (a) the initial state 1 given in Table 5.29 and (b) the steady state 1 given in Table 5.30. The results are shown in Figure 5.47, where it can be seen that starting with both initial states the reactor reaches the high-conversion stable steady state 1.

2. Case II. Two initial states were considered: (a) the initial state 2 given in Table 5.29 and (b) the steady state 2 given in Table 5.30. The results are shown in Figure 5.48, where it can be seen that starting with the initial state 2 (solid line) the reactor reaches the high-conversion stable steady state; while starting with the steady state 2 (dash line) the reactor remains for some time (about 8 hours) at the (unstable) steady state but afterwards it reaches the low-conversion stable steady state.

3. Case III. Two initial states were considered: (a) the initial state 3 given in Table 5.29 and (b) the steady state 3 given in Table 5.30. The results are shown in Figure 5.49, where it can be seen that starting with both initial states the reactor reaches the low-conversion stable steady state 3.

From a practical point of view, the high-conversion steady state is not desired because it implies difficulties in the reactor operation (i.e. high temperatures may produce a reactor run-away); nor the low-conversion steady state because of a low production. Hence the middle-conversion steady state will be the desired set-point. Moreover these results show that a controller must be implemented in order to keep the reactor at the (unstable) middle-conversion steady state.

5.2.3.1 Linear Open-Loop Analysis.

In order to design and implement a linear control, firstly a linear model must be developed and the corresponding linear dynamic behaviour should be analysed and compared to its corresponding nonlinear dynamic behaviour.

Linear dynamic behaviour: Model development.
The linearization of the unperturbed nonlinear model [Eq. 5.164] is as follows:

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}\left(\mathbf{x}_{ss}, \mathbf{u}_{ss}, \mathbf{d}_{ss}, t\right) + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_{ss},\mathbf{u}_{ss},\mathbf{d}_{ss}} \cdot \left(\mathbf{x} - \mathbf{x}_{ss}\right) \tag{5.166}$$

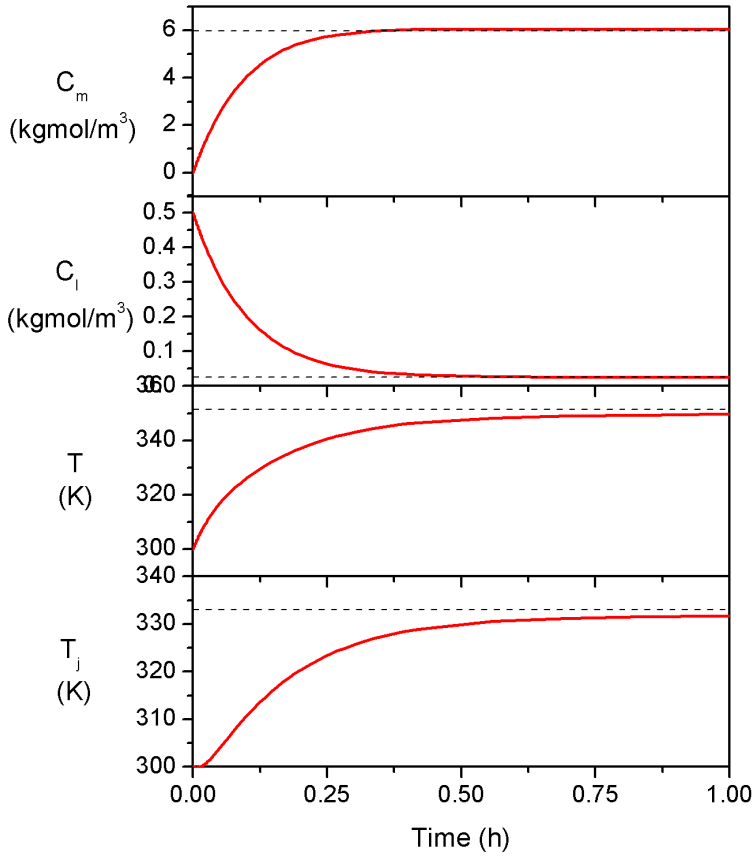


Figure 5.47: Dynamic response of Case I: State 1 (Table 5.29) and steady state 1 (Table 5.30) as initial conditions.

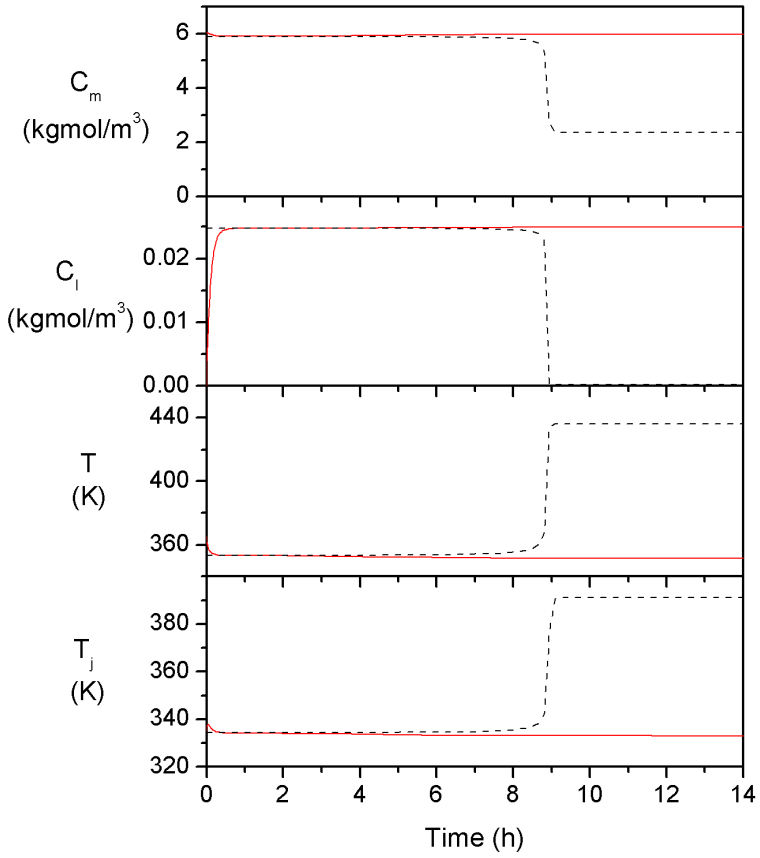


Figure 5.48: Dynamic response of Case II: State 2 (Table 5.29) and steady state 2 (Table 5.30) as initial conditions.

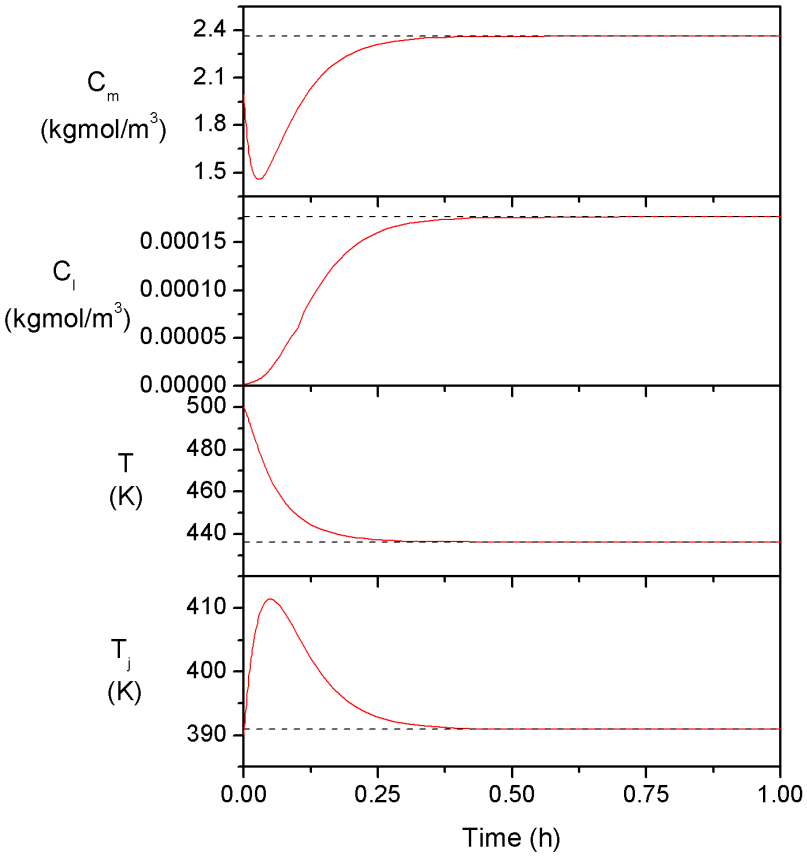


Figure 5.49: Dynamic response of Case III: State 3 (Table 5.29) and steady state 3 (Table 5.30) as initial conditions.

If the linearisation is done around the unstable steady state (steady state 2 of Table 5.30) then the following linear system is obtained:

$$\frac{d\mathbf{x}}{dt} = \mathbf{A} \cdot (\mathbf{x} - \mathbf{x}_{ss}) \tag{5.167}$$

where

$$A := \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_{ss},\mathbf{u}_{ss},\mathbf{d}_{ss}} = \begin{bmatrix} -10.9832 & -117.062 & -0.453582 & 0 \\ 0 & -10.3517 & -0.00107864 & 0 \\ 32.7192 & 3895.61 & -3.24892 & 8.31409 \\ 0 & 0 & 17.1429 & -25.0829 \end{bmatrix} \tag{5.168}$$

Linear dynamic behaviour: Model analysis.
In order to solve the dynamic linear model, the DAE system [equations 5.167] must be solved. The corresponding ICAS-MoT dynamic linear model is given in appendix D. The variable and equation classification of this linear model is as follows:

(a) it has a total of 67 equations, sorted as 63 algebraic and 4 differential equations; and

(b) there are a total of 95 variables, sorted as 27 known, 1 parameter (constant), 63 explicit, and 4 dependent.

The degree of freedom is once again 28, so that the 27 known variables (given in Tables 5.26 - 5.28) and the parameter R must be defined.

Linear dynamic behaviour: Model solution.
The dynamic linear model was also solved using the BDF method, considering two initial states (as Case II for comparison purposes): (a) the initial state 2 given in Table 5.29 and (b) the steady state 2 given in Table 5.30. The results are shown in Figure 5.50, where it can be seen that starting with the initial state 2 (dashed line) the reactor reaches a low-conversion stable steady state (in contrast to the nonlinear behaviour, Figure 5.48, that reached the high-conversion steady state); while starting with the steady state 2 (solid line) the reactor remains at the corresponding steady state (in contrast to the nonlinear behaviour, Figure 5.48, that reached the low-conversion steady state).

These results show that the control, to be designed based on the linear model, must be robust and the tuning must be done carefully to keep the reactor at the set-point.

5.2.3.2 Closed-Loop Analysis

Closed-loop analysis: Linear (perturbed) model.
Considering the nonlinear model given by Eq. 5.164 , the linearization of the

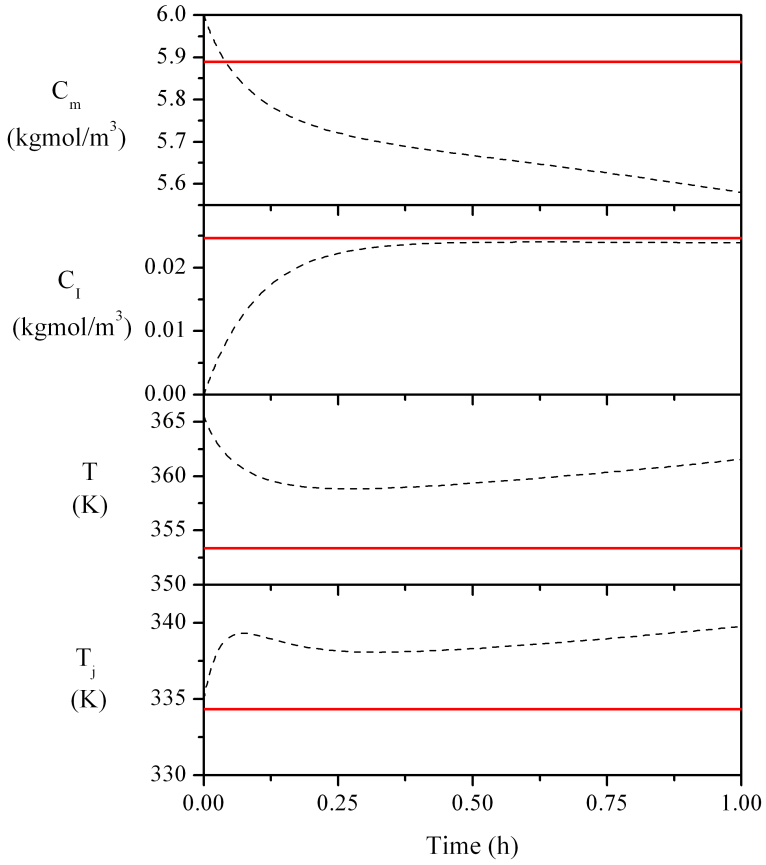


Figure 5.50: Linear dynamic response: State 2 (Table 5.29) and steady state 2 (Table 5.30) as initial conditions.

perturbed model is as follows:

$$\frac{d\mathbf{x}}{dt} = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_{ss},\mathbf{u}_{ss},\mathbf{d}_{ss}} \cdot(\mathbf{x}-\mathbf{x}_{ss}) + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{x}_{ss},\mathbf{u}_{ss},\mathbf{d}_{ss}} \cdot(\mathbf{u}-\mathbf{u}_{ss}) + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{d}}\right|_{\mathbf{x}_{ss},\mathbf{u}_{ss},\mathbf{d}_{ss}} \cdot(\mathbf{d}-\mathbf{d}_{ss}) \tag{5.169}$$

or else

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\cdot(\mathbf{x}-\mathbf{x}_{ss}) + \mathbf{B}\cdot(\mathbf{u}-\mathbf{u}_{ss}) + \mathbf{B}_d\cdot(\mathbf{d}-\mathbf{d}_{ss}) \tag{5.170}$$

where

$$\mathbf{A} := \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_{ss},\mathbf{u}_{ss},\mathbf{d}_{ss}}, \qquad \mathbf{B} := \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{x}_{ss},\mathbf{u}_{ss},\mathbf{d}_{ss}}, \qquad \mathbf{B}_d := \left.\frac{\partial \mathbf{f}}{\partial \mathbf{d}}\right|_{\mathbf{x}_{ss},\mathbf{u}_{ss},\mathbf{d}_{ss}} \tag{5.171}$$

Then the define the deviation variables can be defined

$$\mathbf{x}' = \mathbf{x} - \mathbf{x}_{ss} \tag{5.172}$$

$$\mathbf{u}' = \mathbf{u} - \mathbf{u}_{ss} \tag{5.173}$$

$$\mathbf{d}' = \mathbf{d} - \mathbf{d}_{ss} \tag{5.174}$$

so that the linear perturbed system (Eq. 5.170) can be rewritten as:

$$\frac{d\mathbf{x}'}{dt} = \mathbf{A}\cdot\mathbf{x}' + \mathbf{B}\cdot\mathbf{u}' + \mathbf{B}_d\cdot\mathbf{d}' \tag{5.175}$$

Closed-loop analysis: Control structure.
The control objectives in the polymerisation reactor are to maintain constant values of the outlet monomer concentration C_m (equivalently to monomer conversion) as a measure of quality, the reactor temperature T as a measure of the process safety, by manipulating the initiator feed F_I, and the inlet jacket temperature T_{w0}. In this way, the input ($\mathbf{u}$) and output ($\mathbf{y}$) pairs are as follows:

$$\mathbf{u} = \begin{bmatrix} F_I \\ F_{CW} \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} C_m \\ T_j \end{bmatrix} \tag{5.176}$$

The closed loop reactor and its block diagram are given in Figures 5.51 and 5.52 respectively, and its mathematical formulation is as follows:

$$\begin{aligned} \frac{d\mathbf{x}'}{dt} &= \mathbf{A}\cdot\mathbf{x}' + \mathbf{B}\cdot\mathbf{u}_c^{'} + \mathbf{B}_d\cdot\mathbf{d}' \\ \mathbf{y}' &= \mathbf{C}\cdot\mathbf{x}' \end{aligned} \tag{5.177}$$

where

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{5.178}$$

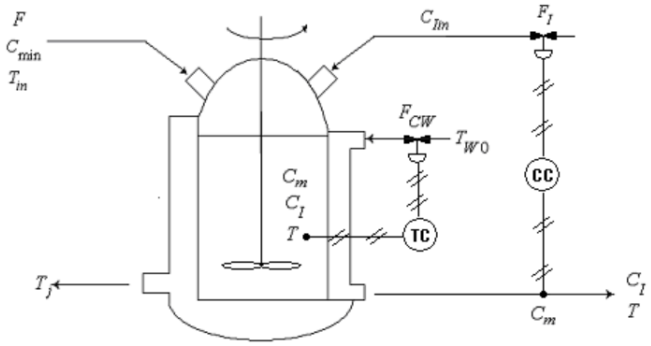


Figure 5.51: Closed-loop polymerisation reactor Flowsheet

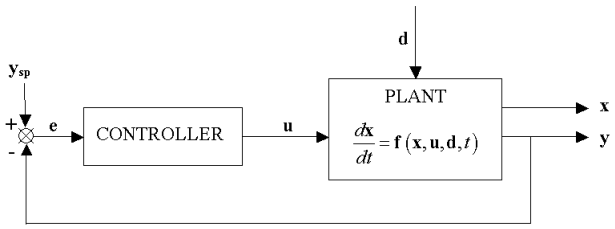


Figure 5.52: Closed-loop Plant

The matrix A evaluated at the unstable steady state is given in Eq. 5.168, and matrices B and B_d are as follows:

$$\mathbf{B} := \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{x}_{ss},\mathbf{u}_{ss},\mathbf{d}_{ss}} = \begin{bmatrix} 0 & 0 & 80 & 0 \\ 0 & 0 & 0 & -2057.85 \end{bmatrix} \tag{5.179}$$

$$\mathbf{B}_d := \left.\frac{\partial \mathbf{f}}{\partial \mathbf{d}}\right|_{\mathbf{x}_{ss},\mathbf{u}_{ss},\mathbf{d}_{ss}} = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 0.032 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 7.94 \end{bmatrix} \tag{5.180}$$

Closed-loop analysis: Controllability matrix.
The linear system given by Eq. (5.177) will be controllable if the controllability matrix

$$\left[\ C = \mathbf{B} \quad \mathbf{AB} \quad \mathbf{A}^2\mathbf{B} \quad \ldots \quad \mathbf{A}^{n-1}\mathbf{B}\ \right] \tag{5.181}$$

has full rank, this is, if Rank C = n (where n = dim[x]).

Using the matrices A [Eq. (5.168)] and B [Eq. (5.179)] evaluated at the unstable steady state, the following controllability matrix is obtained:

$$\left[\ C = \mathbf{B} \quad \mathbf{AB} \quad \mathbf{A}^2\mathbf{B} \quad \mathbf{A}^3\mathbf{B}\ \right] \tag{5.182}$$

Which has *Rank* $C = 4$ (where n = 4, number of states of the polymerisation process). Therefore the polymerisation reactor is controllable.

Closed-loop analysis: PI-Control development.
Implementing a standard PI-controller, the closed loop system is given by

$$\begin{aligned} &\tfrac{d\mathbf{x}'}{dt} = \mathbf{A}\cdot\mathbf{x}' + \mathbf{B}\cdot\mathbf{u}_c' + \mathbf{B}_d\cdot\mathbf{d}' \\ &\mathbf{y}' = \mathbf{C}\cdot\mathbf{x}' \\ &\mathbf{u}_c' = \mathbf{K}_c\cdot\mathbf{e}' + \mathbf{K}_c\cdot\tau_I^{-1}\int_0^t \mathbf{e}'(\tau)\cdot d\tau \end{aligned} \tag{5.183}$$

where

$$\mathbf{u}_c' = \begin{bmatrix} {F_{I,c}}' \\ {F_{CW,c}}' \end{bmatrix}, \quad \mathbf{e}' = \begin{bmatrix} C_m - C_{m,sp} \\ T_j - T_{sp} \end{bmatrix} \tag{5.184}$$

Here $C_{m,sp}$ and T_{sp} are the set point values, this is the values of the unstable steady state.

The gain matrix $\mathbf{K}_c$ and the integral-time matrix τ_I

$$\mathbf{K}_c = \begin{bmatrix} k_{c1} & 0 \\ 0 & k_{c1} \end{bmatrix}, \qquad \tau_I = \begin{bmatrix} \tau_{I1} & 0 \\ 0 & \tau_{I2} \end{bmatrix} \tag{5.185}$$

correspond to the tuning parameters.

Closed-loop analysis: Model analysis. The closed-loop model represented by the DAE system given by equations 5.183]-5.185] was written in ICAS-MoT (see appendix D). The variable and equation classification of this closed-loop model is as follows:

(a) it has a total of 133 equations, sorted as 127 algebraic and 6 differential equations; and

(b) there are a total of 165 variables, sorted as 27 known, 5 parameter (constant), 127 explicit, and 6 dependent.

The degree of freedom is once again 32, so that the 27 known variables (given in Tables 5.26 - 5.28) and the 5 parameters. Where one parameter is R (universal gas constant) and the other 4 correspond to the tuning parameters (two proportional-gains k_{c1}, k_{C2}, and two integral-time constants tau_{I1}, tau_{I2}).

Closed-loop analysis: Model solution.
The closed-loop model was also solved using the BDF method. The tuning parameter values used in the simulations were set as:

$$k_{c1} = k_{c2} = 0.011/h$$

$$\tau_{I1} = \tau_{I2} = 0.5h$$

The steady state 2 given in Table 5.29 was set as the process setpoint. Two cases were simulated to test the robustness of the PI-controller:

Case A. Closed-loop response under a step change in the feed concentration of monomer, $C_{m,in}$ (see Figure 5.53). Results are shown in Figure 5.54, where it can be seen that the set-points for the main objectives, C_m and T, are reached, even when the drastic step changes were applied. As expected, the states C_I and T_J present slower responses, because they are uncontrollable states.

Case B. Closed loop response under a step change in the inlet cool jacket temperature, T_{w0} (see Figure 5.55). Here the setpoints for the main objetives, C_m and T, are reached fastly, and the uncontrollable states C_I and T_J have moderate responses but not slow (meaning that the disturbance T_{w0} has less influence on the process than the disturbance $C_{m,in}$).

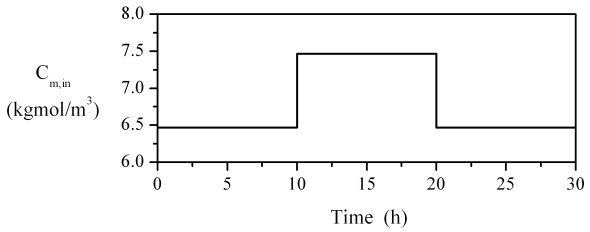


Figure 5.53: Case A. Step change in $C_{m,in}$

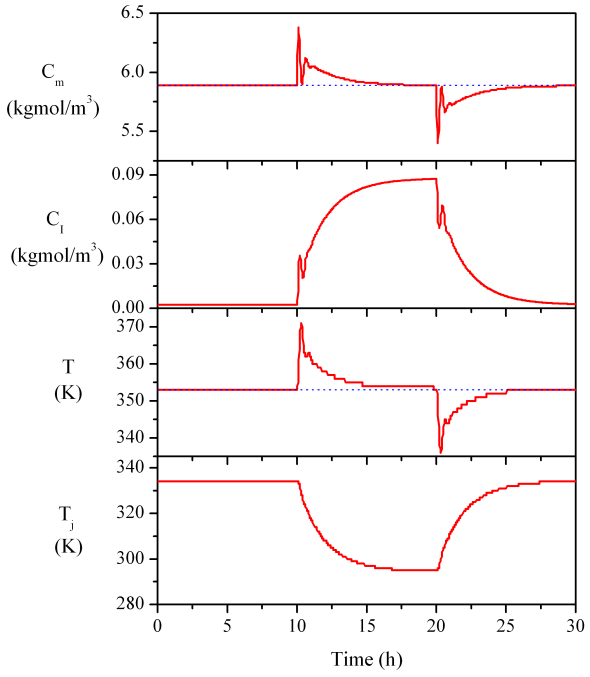


Figure 5.54: Case A. Closed loop response under a step change in the disturbance $C_{m,in}$

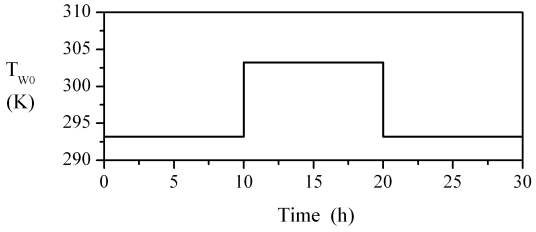


Figure 5.55: Case B. Step change in T_{w0}

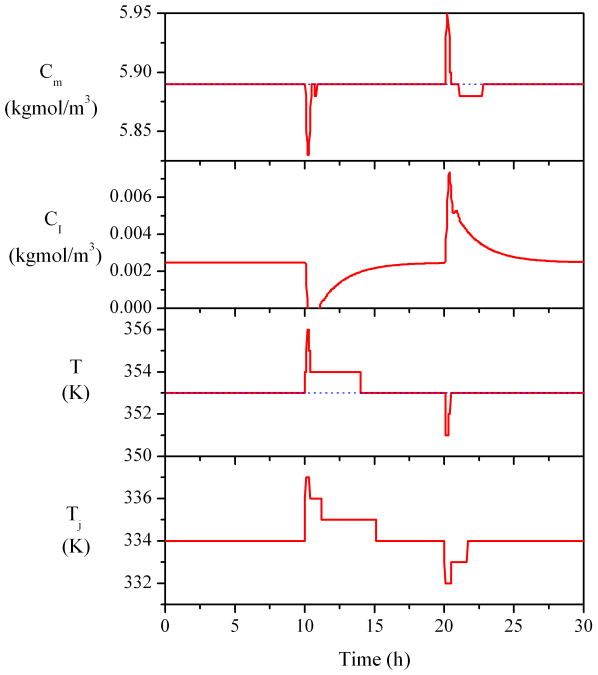


Figure 5.56: Case B. Closed loop response under a step change in the disturbance T_{w0}

Case summary
This example highlighted how ICAS-MoT very easily and quickly permits the simulation of different models and/or process configurations in the same environment, since a core model can be built and gradually expanded during the process life-cycle generating and passing the results from one stage to another. Specifically, in this example we started with (1) the open-loop nonlinear analysis (steady-state, multiplicity of steady states, and dynamic simulation), followed by (2) the open-loop linear analysis (steady-state and dynamic simulations), and finally (3) the closed-loop analysis (dynamic simulations under several step changes in disturbances).

5.2.4 Open and closed loop Plant-wide simulation: The Tennessee Eastman Challenge Problem

◇ **Step 1.***System description*
The Tennessee Eastman Problem (TE) first appeared in an *AIChE* meeting (Downs and Vogel 1990) and at the Chemical Process Control Conference in 1991 since then it has led to a number of publications and research around it. This is a good example of what plant-wide simulation and control issues including.

Figure 5.57 shows the tools that are used to implement and solve the simulation of the open and closed loop of the model plat, in ICAS-MoT. This problem deals with a set of DAE equation system.

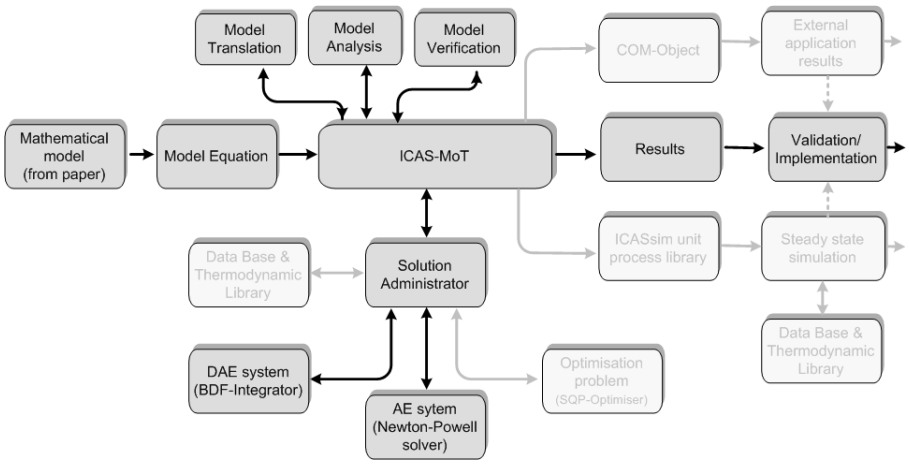


Figure 5.57: Work-Flow and tools used from ICAS-MoT for dynamic simulation (case study 4)

The work-flow that should be followed to solve the problem is highlighted with dark grey colour, whereas the blocks in light grey means that the path is inactivate for problems such as the one tackled here. This work-flow involves the main steps to set up the DAE model and apply the BDF solver.

◇ **Step 2.***Problem definition*
The process produces two products from four reactants. Also present are an inert and a byproduct, making a total of eight components in the system: A, B, C, D, E, F, G, and H. The original reaction scheme is:

$$
\begin{aligned}
A(g) + C(g) + D(g) &\longrightarrow G(l); \quad \text{Product 1} \\
A(g) + C(g) + E(g) &\longrightarrow H(l); \quad \text{Product 2} \\
A(g) + E(g) &\longrightarrow F(l); \quad \text{Byproduct} \\
3D(g) &\longrightarrow 2F(l); \quad \text{Byproduct}
\end{aligned}
$$

All the reactions are irreversible and exothermic. The reaction rates are functions of temperature and represented through an Arrhenius expression. The reaction to produce G has a higher activation energy resulting in higher sensitivity to temperature. All the reactions are approximately first-order with respect to the reactant concentrations.

The process has five major unit operations: a reactor, a product condenser, a vapour-liquid separator, a recycle compressor and a product stripper. Figure 5.58 shows a flow diagram of the process.

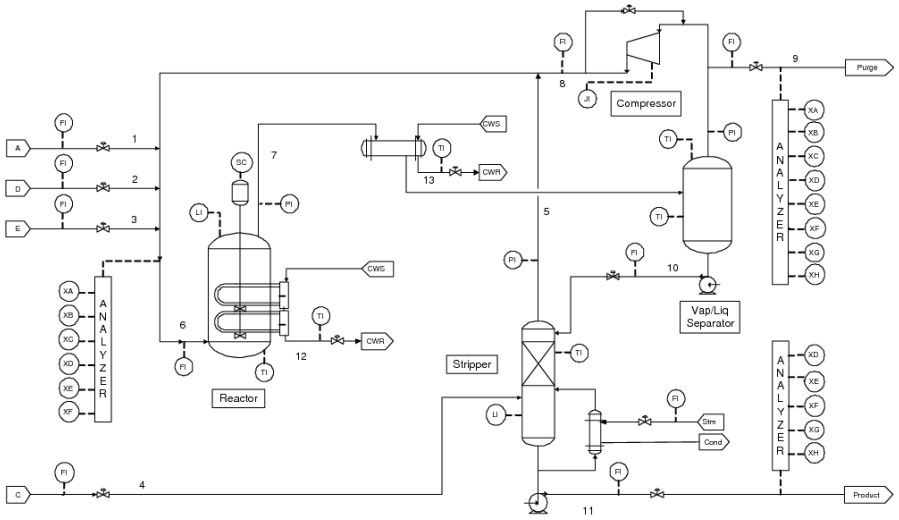


Figure 5.58: TE-Problem flowsheet.

The gaseous reactants are fed to the reactor where they react to form liquid products. The gas phase reactions are catalyzed by a non-volatile catalyst dissolved in the liquid phase. A Continuous Stirred Tank Reactor (CSTR) has an internal cooling bundle for removing the heat evolving from the reaction. The products leave the reactor as vapour, along with the unreacted reactants. The catalyst remains in the reactor.

The reactor product stream passes through a condenser and from there to a vapour-liquid separator. Non-condensed components are recycled back through a centrifugal compressor to the reactor feed. Condensed components move to

a product stripping column to remove the remaining reactants by stripping with feed stream number 4. Products G and H exit the stripper base and are separated in a downstream refining section which is not included in the problem. The inert and byproduct are mainly purged from the system as vapour from the vapour-liquid separator.

The reaction kinetics are as follows:

$$R_1 = \epsilon_1 V_{v,r} \exp\left[44.06 - \frac{42600}{R_g T_r}\right] p_{A,r}^{1.08} p_{C,r}^{0.311} p_{D,r}^{0.874} \quad (5.186)$$

$$R_2 = \epsilon_2 V_{v,r} \exp\left[10.27 - \frac{19500}{R_g T_r}\right] p_{A,r}^{1.15} p_{C,r}^{0.370} p_{E,r}^{1.00} \quad (5.187)$$

$$R_3 = \epsilon_3 V_{v,r} \exp\left[59.50 - \frac{59500}{R_g T_r}\right] p_{A,r}(0.77 p_{D,r} + p_{E,r}) \quad (5.188)$$

where the byproduct reactions have been added in order to produce Eq. (5.188). Therefore, the modified scheme of reaction is:

$$\begin{aligned} A(\mathrm{g}) + C(\mathrm{g}) + D(\mathrm{g}) &\longrightarrow G(\mathrm{l});\ \text{Product 1} \\ A(\mathrm{g}) + C(\mathrm{g}) + E(\mathrm{g}) &\longrightarrow H(\mathrm{l});\ \text{Product 2} \\ \nu A(\mathrm{g}) + D(\mathrm{g}) + \nu E(\mathrm{g}) &\longrightarrow F(\mathrm{l});\ \text{Byproduct} \end{aligned}$$

where $\nu = 1/3$.

◇ **Step 3.** *Mathematical model*

The following main assumptions are made:

$\mathcal{A}_1$. *Mixing zone.* All components are in the vapour phase, therefore the model includes the mass and energy balance equations plus two algebraic equations for mixing zone pressure and composition.

$\mathcal{A}_2$. *Reactor.* The reactor contains liquid and vapour phases, which are assumed to be in equilibrium. There is a significant liquid hold-up of G and H in the reactor, but there is no liquid effluent. To control liquid accumulation in the reactor, one must balance production (by reaction) against vaporization and removal in the gaseous effluent (stream 7) (Downs and Vogel 1993). The feed and the product streams of the reactor are in vapour phase.

$\mathcal{A}_3$. *Product separator.* The deviation from ideality in the vapour-liquid equilibrium is described by a constant activity coefficient for each (condensable) component. In general, where liquid and vapour co-exist, the accumulation of the less volatile components (D through H) in the vapour is neglected. In other words, these components have an equilibrium partial pressure, but the corresponding number of moles in the vapour is neglected when computing the molar hold-ups. Otherwise, an iterative flash calculation would be needed (Ricker and Lee 1995).

$\mathcal{A}_4$. *Compressor and purge.* Ricker and Lee (1995) introduce the recycle stream as an independent variable. This avoids modelling the compressor in detail and the stream flowrate no longer depends on the valve position in the compressor recycle. Similar to the compressor simplification, the purge stream (F_9) is an algebraic variable.

$\mathcal{A}_5$. *Stripper.* There is little information available about the stripper. The unit is characterized by a split fraction model as in Ricker and Lee (1995). Given that the energy balance is added, split fractions (Φ_i) are modelled as third-degree polynomials in temperature.

The mathematical model given by the mass and energy equations in each unit is taking bases in the model equations given by Jockenhövel et al. (2003).

▶ *Mixing zone.* Within the mixing zone all feed streams and the recycle stream are mixed and fed into the reactor.

Molar balances for components *A*–*H*

$$\frac{dN_{i,m}}{dt} = \sum_{j=1,2,3,5,8} y_{ij}F_j - y_{i,6}F_6 \quad (i = A, B, \ldots, H) \tag{5.189}$$

Energy balance for the mixing zone

$$\left(\sum_{i=A}^{H} N_{i,m}Cp_i^v\right)\frac{dT_m}{dt} = \sum_{j=1,2,3,5,8} F_j\left(\sum_{i=A}^{H} y_{i,j}Cp_i^v\right)(T_j - T_m) \tag{5.190}$$

Pressure and concentrations in the mixing zone

$$y_{i,6} = \frac{N_{i,m}}{\sum_{j=A}^{H} N_{j,m}} \tag{5.191}$$

$$P_m = \sum_{i=A}^{H} N_{i,m}\frac{R_g T_m}{V_m} \tag{5.192}$$

▶ *Reactor.* In the reactor model the influence of the agitator is neglected and excess heat is removed by cooling water. Feed and product streams are in vapour phase.

Molar balances for components *A*-*H*

$$\frac{dN_{i,r}}{dt} = y_{i,6}F_6 - y_{i,7}F_7 + \sum_{k=1}^{3}\nu_{i,k}R_k; \qquad i = A, \ldots H \tag{5.193}$$

Energy balance for the reactor

$$\left(\sum_{i=A}^{H} N_{i,r} c_{p,i}\right) \tfrac{dT_r}{dt} = F_6 \left(\sum_{i=A}^{H} y_{i,6} c_{p,vap,i}\right)(T_6 - T_r) - \dot{Q}_r - \sum_{k=1}^{3} \Delta H_{rk} R_k \tag{5.194}$$

$$\Delta H_{r,k} = \sum_{i=A}^{H} H_i \nu_{i,k} + HoF_k, \quad \text{with} \quad H_i = Cp_i^v (T_r - T_{ref}) \tag{5.195}$$

where the reaction kinetics are

$$\widehat{R}_1 = y_{A,3}^{1.08}\, y_{C,3}^{0.311}\, y_{D,3}^{0.874} \tag{5.196a}$$

$$\widehat{R}_2 = \frac{\epsilon_2 \exp\left(c_{1,2} - c_{2,2}/R_g T_r\right)}{\epsilon_1 \exp\left(c_{1,1} - c_{2,1}/R_g T_r\right)} P_r^{2.52}\, y_{A,3}^{1.15} y_{C,3}^{0.370} y_{E,3}^{1.00} \tag{5.196b}$$

$$\widehat{R}_3 = \frac{\epsilon_3 \exp\left(c_{1,3} - c_{2,3}/R_g T_r\right)}{\epsilon_1 \exp\left(c_{1,1} - c_{2,1}/R_g T_r\right)} P_r^{2}\, y_{A,3}(0.77 y_{D,3} + y_{E,3}) \tag{5.196c}$$

Heat exchange with cooling water

$$\dot{Q}_r = m_{cw,r} Cp_{cw} (T_{cw,r,out} - T_{cw,r,in}) \tag{5.197}$$

$$\dot{Q}_r = UA_r \left(\frac{\Delta T_{1,r} - \Delta T_{2,r}}{\ln(\Delta T_{1,r}/\Delta T_{2,r})}\right) \tag{5.198}$$

$$\Delta T_{1,r} = T_r - T_{cw,r,in}; \qquad \Delta T_{2,r} = T_r - T_{cw,r,out} \tag{5.199}$$

Vapour-liquid equilibrium

$$p_{i,r} = \gamma_{i,r}\, x_{i,r}\, p_{i,r}^{sat}(T_r); \qquad i = D, \ldots H \tag{5.200}$$

$$p_{i,r} = \frac{N_{i,r}^v\, R_g T_r}{V_{v,r}}; \qquad i = A, B, C \tag{5.201}$$

$$p_{i,r}^{sat}(T) = 10^{-3} \exp\left(A_i + \frac{B_i}{C_i + T_r - T_{ref}}\right); \quad i = D, \ldots H \tag{5.202}$$

$$P_r = \sum_{i=A}^{H} p_{i,r} \tag{5.203}$$

$$y_{i,7} = \frac{p_{i,r}}{P_r}; \qquad i = A, \ldots H \tag{5.204}$$

$$x_{i,r} = \frac{N_{i,r}}{\sum_{i=D}^{H} N_{i,r}}; \qquad i = D, \ldots H \tag{5.205}$$

$$V_{l,r} = \frac{\sum_{i=D}^{H} N_{i,r}^l}{\rho_{l,r}}, \quad \text{with} \quad V_{v,r} = V_r - V_{l,r} \tag{5.206}$$

Reactor input stream F_6 and reactor output stream F_7

$$F_6 = 0.8334 \frac{\text{kmol}}{\text{s}\sqrt{\text{MPa}}} \sqrt{P_m - P_r} \tag{5.207}$$

$$F_7 = 1.5355 \frac{\text{kmol}}{\text{s}\sqrt{\text{MPa}}} \sqrt{P_r - P_s} \tag{5.208}$$

The constant values in Eqs. (5.207) and (5.208) are given by Jockenhövel et al. (2003) to match the base case. Note that the cooling water flux $m_{cw,r}$ is a control variable directly. Its dependence on the valve position is neglected. Note that for components *D*-*H* the numbers of moles $N_{i,r}$ refer to the number of moles in the liquid phase only, due to the assumption that build-up of these components in the vapour phase can be neglected.

▶ *Product Separator.* Molar balances for components A-H

$$\frac{dN_{i,s}}{dt} = y_{i,7}F_7 - y_{i,8}(F_8 + F_9) - x_{i,10}F_{10}; \quad i = A, \dots H \tag{5.209}$$

Energy balance for the separator

$$\left(\sum_{i=A}^{H} N_{i,s} Cp_i \right) \frac{dT_s}{dt} = F_7 \left(\sum_{i=A}^{H} y_{i,7} Cp_i^v \right) (T_r - T_s) - H_0 V_s - \dot{Q}_s \tag{5.210}$$

$$HoV_s = \sum_{i=D}^{H} x_{i,10} F_{10} H_{vap,i} \tag{5.211}$$

Vapour-liquid equilibrium

$$p_{i,s} = \frac{N_{i,s}^v \, R_g T_s}{V_{v,s}}; \quad i = A, B, C \tag{5.212}$$

$$P_s = \sum_{i=A}^{H} p_{i,s} \tag{5.213}$$

$$p_{i,s} = \gamma_{i,s} \, x_{i,10} \, p_{i,s}^{sat}(T_s); \quad i = D, \dots H \tag{5.214}$$

$$y_{i,8} = y_{i,9} = \frac{p_{i,s}}{P_s} \tag{5.215}$$

$$x_{i,10} = 0; \quad i = A, B, C; \quad x_{i,10} = \frac{N_{i,s}}{\sum_{i=D}^{H} N_{i,s}}; \; i = D, \dots H \tag{5.216}$$

$$V_{l,s} = \frac{\sum_{i=D}^{H} N_{i,s}^l}{\rho_{l,s}}; \quad V_{v,s} = V_s - V_{l,s} \tag{5.217}$$

Heat exchange with cooling water

$$\dot{Q}_s = m_{cw,s} Cp_{cw} (T_{cw,s,out} - T_{cw,s,in}) \tag{5.218}$$

$$\dot{Q}_s = UA_s \left(\frac{\Delta T_{1,s} - \Delta T_{2,s}}{\ln \Delta T_{1,s} / \Delta T_{2,s}} \right) \tag{5.219}$$

$$\Delta T_{1,s} = T_s - T_{cw,s,in}; \qquad \Delta T_{2,s} = T_s - T_{cw,s,out} \tag{5.220}$$

► *Compressor and purge.* The temperature changes due to the compressor work are taken into account by the Eq. (5.221).

$$T_8 = T_s \left(\frac{P_m}{P_s} \right)^{\frac{1-\kappa}{\kappa}} \tag{5.221}$$

► *Stripper.* This unit is modelled by a split fraction model as proposed in Ricker and Lee (1995). An energy balance was added by Jockenhövel et al. (2003) and the split fractions (Φ_i) are modelled as third-degree polynomials in temperature. The pressures in the stripper and the mixing zone are assumed to be the same. The heating medium is saturated steam, which condenses completely at a constant temperature. The enthalpy of the steam has been chosen to fit the steam flux with the heat duty of the stripper given in Downs and Vogel (1993) for the base case.

Molar balances for components *G*-*H*

$$\frac{dN_{i,p}}{dt} = (1 - \Phi_i)(x_{i,10} F_{10} + y_{i,4} F_4) - x_{i,11} F_{11}; \quad i = G, H \tag{5.222}$$

Energy balance for the stripper

$$\begin{aligned} \left(\sum_{i=G}^{H} N_{i,p} Cp_i \right) \tfrac{dT_p}{dt} = F_{10} \left(\sum_{i=A}^{H} x_{i,10} Cp_i \right) (T_s - T_p) + \\ F_4 \left(\sum_{i=A}^{H} y_{i,4} Cp_i^v \right) (T_4 - T_p) - H_0 V_p + \dot{Q}_p \end{aligned} \tag{5.223}$$

$$HoV_{str} = \sum_{i=D}^{H} H_{vap,i} \left(y_{i,5} F_5 - y_{i,4} F_4 \right) \tag{5.224}$$

$$\dot{Q}_{str} = 2258.717 \frac{\text{kJ}}{\text{kg}} \dot{m}_{cw,str} \tag{5.225}$$

Vapour-liquid equilibrium

$$V_{l,str} = \sum_{i=D}^{H} \frac{N_{i,str}}{\rho_{str}} \tag{5.226}$$

$$\Phi_i = 1; \qquad i = A, B, C; \tag{5.227}$$

$$\Phi_i = \sum_{j=0}^{3} a_{i,j}(T_s - 273)^j; \quad i = D, \ldots H \tag{5.228}$$

$$F_5 = F_{10} + F_4 - F_{11} \tag{5.229}$$

$$y_{i,5} = \frac{\Phi_i(y_{i,4}F_4 + x_{i,10}F_{10})}{F_5}; \qquad i = A, \ldots H \tag{5.230}$$

$$x_{i,11} = \frac{y_{i,4}F_4 + x_{i,10}F_{10} - y_{i,5}F_5}{F_{11}}; \qquad i = D, \ldots, F; \tag{5.231}$$

$$x_{i,11} = \left(1 - \sum_{j=D}^{F} x_{j,11}\right) \frac{N_{i,str}}{\sum_{j=D}^{H} N_{j,str}}; \quad i = G, H \tag{5.232}$$

◇ **Step 4/5.** *Model analysis and data needed.*
In order to solve the reactor model dynamically, the DAE system must be solved simultaneously. The corresponding ICAS-MoT dynamic model is given in appendix D. The variable and equation classification of this model, obtained through ICAS-MoT is as follows:

With the modifications made by Jockenhövel et al. (2003), and included in the model used in here there are 30 states, and 11 manipulated variables.

The model equations (5.189)–(5.232) can be presented in a vectorial (compact) form, thus

$$\dot{\mathbf{x}} := \frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{p}) \tag{5.233}$$

$$\mathbf{y} = \mathbf{g}(\mathbf{x}, \mathbf{u}, \mathbf{p}) \tag{5.234}$$

$$\mathbf{x} = [\mathbf{x_s}, \mathbf{x_p}] \qquad \text{where}$$

$$\mathbf{x_s} = [T_m, T_r, T_s, T_{str}, N_{i,m}, N_{i,r}, N_{i,s}, N_{G,str}, N_{H,str}]^T;\ i = A, \ldots H \tag{5.235a}$$

$$\begin{aligned}\mathbf{x_p} = [&F_5, F_6, F_7, P_m, P_r, P_s, T_8, T_{cw,r,out}, T_{cw,s,out}, V_{l,r}, V_{v,r}, V_{l,s}, V_{v,s}, V_{l,str},\\ &H_i, HoV_s, HoV_{str}, \dot{Q}_r, \dot{Q}_s, \dot{Q}_{str}, \Delta T_{1,r}, \Delta T_{2,r}, \Delta T_{1,s}, \Delta T_{2,s},\\ &p_{i,r}, p_{i,r}^{sat}, p_{i,s}, p_{i,s}^{sat}, x_{i,r}, x_{i,10}, x_{i,11}, y_{i,5}, y_{i,6}, y_{i,7}, y_{i,8}, y_{i,9},\\ &R_k, \Delta H_{r,k}, \Phi_i,]^T; \qquad i = A, \ldots H,\ k = 1 \ldots NR\end{aligned} \tag{5.235b}$$

$$\mathbf{u} = [F_1, F_2, F_3, F_4, F_8, F_9, F_{10}, F_{11}, m_{cw,r}, m_{cw,s}, m_{cw,str}]^T \tag{5.236}$$

$$\begin{aligned}\mathbf{p} = [&T_1, T_2, T_3, T_4, T_{cw,r,in}, T_{cw,s,in}, V_m, V_r, V_s, UA_r, UA_s, R_g, \kappa, \rho_r, \rho_s, \rho_{str},\\ &A_i, B_i, C_i, Cp_i^l, Cp_i^v, Cp_{cw}, HoF_k, H_{vap,i}, x_{i,1}, x_{i,2}, x_{i,3}, x_{i,4}, \gamma_{i,r}, \gamma_{i,s},\\ &\epsilon_k, a_{c,j}]^T;\ c = D, \ldots H, i = A, \ldots H,\ j = 0, \ldots 3,\ k = 1, \ldots NR\end{aligned} \tag{5.237}$$

The state vector $\mathbf{x}$ is comprised of 30 state variables ($\mathbf{x_s}$) and 129 process variables ($\mathbf{x_p}$); vector $\mathbf{u}$ contains 11 design variables, and 100 properties and parameters of the system listed in vector $\mathbf{p}$

♦ **Model data.**

Table 5.32 lists the physical properties for the components in the TE-problem.

Table 5.32: Components physical properties (Downs and Vogel 1993).

Component	Molecular weight	Liquid density (kg/m^3)	Liquid heat capacity (kJ/kg/°C)	Vapour heat capacity (kJ/kg/°C)	Heat of vaporization (kJ/kg)
A	2.0	–	–	14.6	–
B	25.4	–	–	2.04	–
C	28.0	–	–	1.05	–
D	32.0	299.0	7.66	1.85	202
E	46.0	365.0	4.17	1.87	372
F	48.0	328.0	4.45	2.02	372
G	62.0	612.0	2.55	0.712	523
H	76.0	617.0	2.45	0.628	486

Vapour Pressure (Antoine equation):
$P = \exp\left[A_i + B_i/(C_i + T(°C))\right]$

Component	Constant A	Constant B	Constant C
D	20.81	−1444.0	259
E	21.24	−2114.0	266
F	21.24	−2144.0	266
G	21.32	−2748.0	233
H	22.10	−3318.0	250

Parameters values for components A, B and C are not listed because they are effectively noncondensible

In Table 5.33 are listed the stream conditions utilised at the base case scenario (Downs and Vogel 1993). Molar compositions and temperature do not change between operation modes.

Table 5.34 gives the parameter values used in Eqns. (5.196a)–(5.196c).

The parameter specifications (vector $\mathbf{p}$) are given in Table 5.35.

◇ **Step 6.** *Model solution.*

♦ **Steady-state solution.**

In order to solve the process model in steady-state, the transient term in equations (5.189)–(5.232) must be equal to zero. The corresponding ICAS-MoT steady-state model is given in appendix D.

Three different operation modes were analysed in this process. Table 5.36 lists the manipulated variables (vector $\mathbf{u}$) corresponding to the these modes (Ricker

Table 5.33: Stream conditions at base case (Downs and Vogel 1993).

	Mole fraction $y_{i,j}$ in streams			
Component	1	2	3	4
A	0.99990	0.00000	0.00000	0.48500
B	0.00010	0.00010	0.00000	0.00500
C	0.00000	0.00000	0.00000	0.51000
D	0.00000	0.99990	0.00000	0.00000
E	0.00000	0.00000	0.99990	0.00000
F	0.00000	0.00000	0.00010	0.00000
G	0.00000	0.00000	0.00000	0.00000
H	0.00000	0.00000	0.00000	0.00000
Flow (kmol/h)=	11.2	114.5	98.0	417.5
T (°C) =	45.0	45.0	45.0	45.0

Table 5.34: Kinetic values in rate expressions.

$c_{1,1}$ =	44.06	$c_{2,1}$ =	42600
$c_{1,2}$ =	10.27	$c_{2,2}$ =	19500
$c_{1,3}$ =	59.50	$c_{2,3}$ =	59500

and Lee 1995).

Table 5.37 summarises the 23 outputs included in y. The last column in Table 5.37 gives the corresponding output reported by Downs and Vogel's (1993).

In this respect, the results obtained for these operation modes in terms of streams compositions and flowrates are given in Tables 5.38–5.40.

♦Model modification
The model above is now used in its dynamic form. This will highlight that the same ICAS-MoT code with a small modifications can be used in a different model configurations.

Dynamic simulations This particular case study is characteristic because of its unstable performance in the open-loop. Consequently, it is important to analyse the dynamic behaviour of the system. Dynamic simulations were set up through ICAS-MoT by selecting the BDF method for integration of the DAE system.

$$\mathbf{y} = [F_{11}, P_r, T_r, T_{str}, V_{l,r}, V_{l,s}, V_{l,str}, y_{B,9}, y_{C,9}]^T \tag{5.238}$$

Consequently, the dynamic model has 30 ODEs (states) and 129 AEs with 11 control variables.

Figures 5.59 and 5.60 show the (open-loop) dynamic response for the reactor

Table 5.35: Parameter specifications for TE-Problem (Jockenhövel et al. 2003).

Mixing section			**Reactor**		
V_m	=	141.53 m^3	V_r	=	36.8117791 m^3
R_g	=	8.31451 kPa m^3/kmol/K	$\rho_{l,r}$	=	7.28223 kmol/m^3
Compressor			ϵ_1	=	1.0399157 kmol/h/m^3
κ	=	0.7166374645	ϵ_2	=	1.0113731 kmol/h/m^3
T_{ref}	=	273.15 K	ϵ_2	=	1.00 kmol/h/m^3
			$\gamma_{D,r}$	=	0.996011
Separator			$\gamma_{E,r}$	=	1.0
$\rho_{l,s}$	=	10.29397546 kmol/m^3	$\gamma_{F,r}$	=	1.078
$\gamma_{D,s}$	=	1.001383	$\gamma_{G,r}$	=	0.999
$\gamma_{E,s}$	=	1.001383	$\gamma_{H,r}$	=	0.999
$\gamma_{F,s}$	=	1.001383	$T_{cw,r,in}$	=	308.0 K
$\gamma_{G,s}$	=	1.001383	UA_r	=	127.6 kW/K
$\gamma_{H,s}$	=	0.992188	$m_{cw,r}$	=	93.7 m^3/h
V_s	=	99.1 m^3	Cp_{cw}	=	4.18 kJ/kg/K
$T_{cw,s,in}$	=	313.0 K	HoF_1	=	−136033.04 kJ/kmol
UA_s	=	152.7 kW/K	HoF_2	=	−93337.9616 kJ/mol
$m_{cw,s}$	=	49.37 m^3/h	HoF_3	=	0.0 kJ/mol
Stripper					
$\rho_{l,str}$	=	8.6496 kmol/m^3	$a_{6,2}$	=	−0.00010
$m_{cw,str}$	=	230.31 kg/h	$a_{6,3}$	=	3.69×10^{-7}
$a_{4,0}$	=	0.548012	$a_{7,0}$	=	0.001393
$a_{4,1}$	=	0.011351	$a_{7,1}$	=	0.000217
$a_{4,2}$	=	−0.00011	$a_{7,2}$	=	1.37×10^{-5}
$a_{4,3}$	=	3.51×10^{-7}	$a_{7,3}$	=	1.84×10^{-9}
$a_{5,0}$	=	0.620794	$a_{8,0}$	=	−0.01568
$a_{5,1}$	=	0.010197	$a_{8,1}$	=	0.000976
$a_{5,2}$	=	−0.00010	$a_{8,2}$	=	-7.64×10^{-6}
$a_{5,3}$	=	3.69×10^{-7}	$a_{8,3}$	=	8.32×10^{-8}
$a_{6,0}$	=	0.628854			
$a_{6,1}$	=	0.010049			

Table 5.36: Manipulated variables for operation modes.

Input	Units	Base case	Mode 1 (50/50)	Mode 2 (10/90)
F_1	kmol/h	11.200	11.991	13.848
F_2	kmol/h	114.500	114.314	22.948
F_3	kmol/h	98.000	96.471	174.679
F_4	kmol/h	417.500	413.782	383.109
F_8	kmol/h	1201.500	1441.021	1419.501
F_9	kmol/h	15.100	9.497	16.164
F_{10}	kmol/h	259.500	253.563	243.825
F_{11}	kmol/h	211.885	210.885	194.638
$m_{cw,r}$	m^3/h	93.70	75.40	62.95
$m_{cw,s}$	m^3/h	49.37	52.02	47.40
$m_{cw,str}$	kg/h	230.31	4.74	4.90

Table 5.37: Elements of the output vector (y)

No.	Description	Units	XMEAS Downs & Vogel, (1993)
1	Reactor pressure	kPa	7
2	Reactor liquid level	%	8
3	Separator pressure	kPa	13
4	Separator liquid level	%	12
5	Stripper bottoms level	%	15
6	Stripper pressure	kPa	16
7	Reactor feed flow rate	kscmh	6
8	A in the reactor feed (stream 6)	mol%	23
9	B in the reactor feed	mol%	24
10	C in the reactor feed	mol%	25
11	D in the reactor feed	mol%	26
12	E in the reactor feed	mol%	27
13	F in the reactor feed	mol%	28
14	A in purge (stream 9)	mol%	29
15	B in purge	mol%	30
16	C in purge	mol%	31
17	D in purge	mol%	32
18	E in purge	mol%	33
19	F in purge	mol%	34
20	G in purge	mol%	35
21	H in purge	mol%	36
22	G in product (stream 11)	mol%	40
23	H in product	mol%	41

Table 5.38: Base case results.

	Mole fraction $y_{i,j}$ in streams						
Component	5	6	7	8	9	10	11
A	0.45744	0.31928	0.27178	0.31724	0.31724	0.00000	0.00000
B	0.00448	0.07872	0.10765	0.12249	0.12249	0.00000	0.00000
C	0.43457	0.24844	0.18457	0.22363	0.22363	0.00000	0.00000
D	0.00124	0.07024	0.01030	0.01508	0.01508	0.00242	0.00024
E	0.07211	0.19793	0.18477	0.20286	0.20286	0.13427	0.00598
F	0.01197	0.02431	0.02702	0.03372	0.03372	0.02232	0.00104
G	0.02314	0.04659	0.13179	0.06456	0.06456	0.55138	0.54651
H	0.00630	0.01449	0.08212	0.02042	0.02042	0.28961	0.44623
Flow =	465.7	1892.102	1476.0	1201.5	15.1	259.5	211.3
mass $S_{G/H}$ =	0.999104						

Flowrate [=] kmol/h, mass $S_{G/H} = (y_{G,11} MW_G)/(y_{H,11} MW_H)$

Table 5.39: Mode 1. 50/50 G/H mass ratio results.

	Mole fraction $y_{i,j}$ in streams						
Component	5	6	7	8	9	10	11
A	0.43965	0.32346	0.28053	0.32959	0.32959	0.00000	0.00000
B	0.00453	0.14777	0.18403	0.21655	0.21655	0.00000	0.00000
C	0.46232	0.18947	0.11387	0.13307	0.13307	0.00000	0.00000
D	0.00072	0.05933	0.00587	0.00798	0.00798	0.00140	0.00014
E	0.05666	0.16522	0.15097	0.15885	0.15885	0.10569	0.00443
F	0.01939	0.04123	0.05199	0.05467	0.05467	0.03622	0.00160
G	0.02067	0.04979	0.1291	0.0669	0.0669	0.48395	0.60702
H	0.00826	0.02373	0.08363	0.03239	0.03239	0.37273	0.38682
Flow=	456.46	2125.822	1707.081	1441.021	9.40888	253.563	210.885
mass $S_{G/H}$ =	1.280171						

Flowrate [=] kmol/h, mass $S_{G/H} = (y_{G,11}MW_G)/(y_{H,11}MW_H)$

Table 5.40: Mode 2. 10/90 G/H mass ratio results.

	Mole fraction $y_{i,j}$ in streams						
Component	5	6	7	8	9	10	11
A	0.42958	0.34465	0.30864	0.36252	0.36252	0.00000	0.00000
B	0.00443	0.07869	0.09704	0.11352	0.11352	0.00000	0.00000
C	0.45172	0.19418	0.12464	0.14587	0.14587	0.00000	0.00000
D	0.00005	0.01153	0.00107	0.00059	0.00059	0.00010	0.00001
E	0.08549	0.26240	0.22146	0.23337	0.23337	0.15672	0.00726
F	0.02776	0.05793	0.07249	0.07610	0.07610	0.05097	0.00246
G	0.00399	0.00920	0.02492	0.01222	0.01222	0.09458	0.11842
H	0.01527	0.04142	0.14974	0.05581	0.05581	0.69763	0.87184
Flow=	430.4475	2062.893	1683.037	1413.116	16.09766	243.825	194.638
mass $S_{G/H}$ =	0.110807						

Flowrate [=] kmol/h, mass $S_{G/H} = (y_{G,11}MW_G)/(y_{H,11}MW_H)$

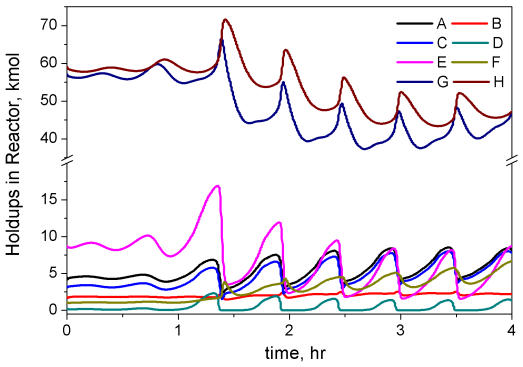


Figure 5.59: Reactor hold-ups profiles for the base case.

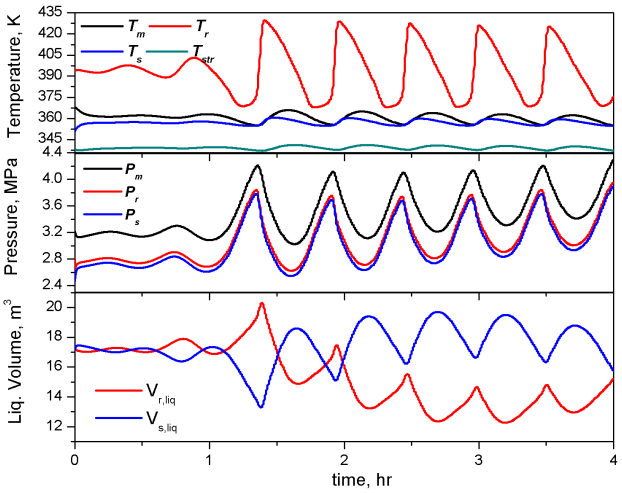


Figure 5.60: Temperature and pressure profiles for the base case.

hold-ups, as well as the temperature, liquid volume and pressure profiles in the equipments, respectively. It is observed that there are small oscillations with increasing amplitude in the profiles during the first hour of operation. After this initial period, the amplitude of the oscillations increases substantially, particularly, in the reactor temperature. Hence, the instability of the process is confirmed because the safety limits has been reached and therefore the operation has to be shut down.

♦ **Closed-loop simulation**
Now, the dynamic model implemented in ICAS-MoT is modified by adding the control equations, the ICAS-MoT model is given in appendix D.

A simple PI-controller was implemented. It has XMEAS_15 = stripper liquid product flow (with a step change of + 15) as measured output (y), and $F11$ = stripper outlet flow rate as control input (u). The control law used is as follows:

Set Point

$$Setpt = 65 \tag{5.239}$$

Error and time-derivative of the error

$$e = Setpt - X_{meas}\,(15) \tag{5.240}$$

$$\frac{de}{dt} = -22.58\left(\frac{1}{\rho_G}\cdot\frac{dN_{G,p}}{dt} + \frac{1}{\rho_H}\cdot\frac{dN_{H,p}}{dt}\right) \tag{5.241}$$

Tuning

$$K_{\mathrm{p}} = \frac{1}{0.3} \tag{5.242}$$

Controller : Stripper liquid product flow %

$$F_{11} = F_{11,reference} - K_{\mathrm{p}}\cdot\left(e + \int\frac{e}{\tau_I}dt\right) \tag{5.243}$$

or else

$$\frac{dF_{11}}{dt} = -K_{\mathrm{p}}\left(\frac{de}{dt} + \frac{e}{\tau_I}\right) \tag{5.244}$$

The simulation was done using the BDF method available in ICAS-MoT. The results are shown in Figures 5.61-5.63, where it can be seen that the control input (u) $F11$ reaches steady-state, while the measured output XMEAS_15 reaches the given *Set Point* = 65. The other outputs (XMEAS_6, XMEAS_8 and XMEAS_12) also remain in a almost constant value (notice the small scale), achieving the control purpose.

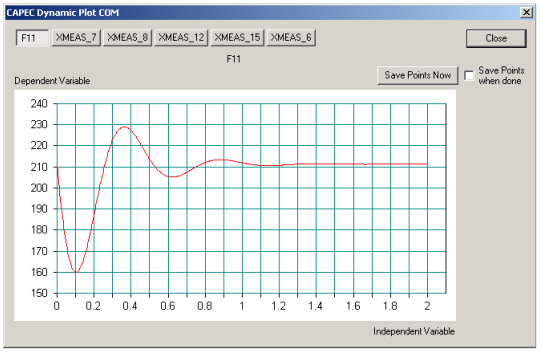


Figure 5.61: Dynamic behaviour of input: F11 = stripper outlet flow rate.

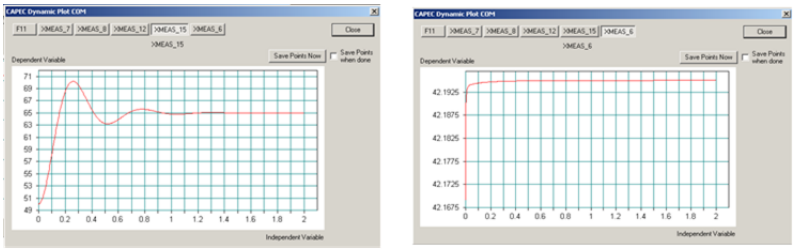


Figure 5.62: Dynamic behaviour of reactor outputs: (a) XMEAS_15 = Stripper level, (b) XMEAS_6 = Reactor feed flow rate.

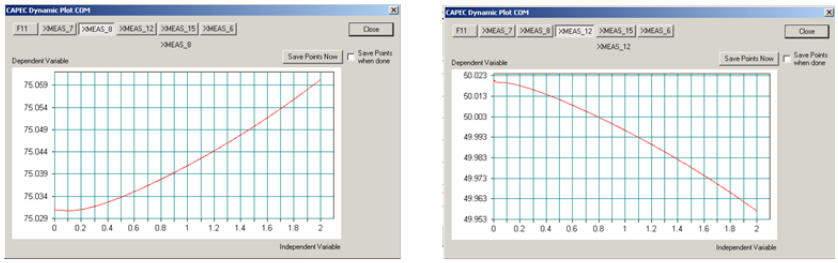


Figure 5.63: Dynamic behaviour of reactor outputs: (c) XMEAS_8 = Reactor liquid level, (d) XMEAS_12 = Separator liquid level.

Case summary.
The application of the process modelling tool ICAS-MoT to large-scale chemical engineering processes was presented. The Tennessee Eastman plant is an interesting problem to highlight how the equation-based models of the different process units can be integrated in ICAS-MoT. Three different simulations modes have been performed (steady-state, open-loop and closed-loop simulations), validating the unstable nature of the TE-problem. It has been verified as well that, as Downs and Vogel (1993) and Ricker and Lee (1995) point out, after the first hour of operation the process starts to become unstable reaching the safety limits, therefore the operation has to be shut down.
The main concepts in this respect are the development of a simplified implementation of sub-models (on the level of process units) and the flexibility to effectively perform dynamic simulation.

5.3 Other ICAS-MoT models implemented

Several, other process operation models that are important in chemical product manufacturing have been implemented in ICAS-MoT, but are not treated further in this thesis. Table 5.41 list some of them.

Table 5.41: Models implemented in ICAS-MoT

Process	Description	Type	Equations	Reference
Membrane Distillation	Vacuum, Direct Contact and Sweeping Gas	DAE	75 explicit, 5 ODE	Lawson and Lloyd, 1996.
Pervaporation	Dynamic Simulation	DAE	5 ODE, 75 explicit	Søren M., 1998.
Emulsion copolymerisation Reactor MMA-S	Open-loop dynamic simulation for different configurations	DAE	4 ODE, 70 explicit	Dimitratos et al., 1989
Solution Copolymerisation Reactor MMA-VA	Open-loop steady-state analysis (multiplicity and stability) and dynamic behavior.	DAE	18 ODE, 77 explicit	Bindlish et al., 2003; Congalidis et al., 1989.
Thermal Cracker	Profit maximization. The variables to be optimised are the feed amounts.	AE-Optimiser	7 explicit and 1 objective function	Edgar and Himmelblau, 1998.
CSTR and PFR (Optim. problems)	Attainable region problems, linear and nonlinear constrain minimization	AE, ODE, DAE	-	Fogler, 1999.
Isothermal Batch Reactor	Calculate the optimal residence time to yield a maximum value of intermediate product.	Nonlinear Constrained Miniminisation	AE	Edgar and Himmelblau, 1998.
Thermodynamic models	Calculation of fugacity/activity using: UNIFAC, Wilson, SRK, SAFT, PC-SAFT, CPA	AE	-	Several authors
Hybrid process	Well stirred tank reactor coupled with nanofiltration separation unit.	DAE	14 ODE and 43 AE	Whu et al. 1999.

6

Conclusions

A computer-aided modelling tool called ICAS-MoT, that can assist the model developer and engineer by carrying out expensive (with respect to time and resources) steps in the modelling process and reducing the overall time consumed, has been presented together with detailed application examples. These examples were related to different model applications for bio- and chemical processes, such as steady state and dynamic simulations, static and dynamic process optimisation studies, model parameter estimation, and development and quick test of new models reported in journals and books. ICAS-MoT has also been used to generate process models very quickly that are currently not available in process simulation packages (e.g. short-path evaporation and special physical property models).

Considerable effort has been put in the development and continuos improvement of ICAS-MoT. It should be recognized, however, that even with a relatively sophisticated general purpose process modelling tool, the modelling and simulation of chemical processes still presents serious difficulties. One key problem is the mathematical complexity of the models: population balances invariably lead to partial differential equations, and these are often coupled with other equations describing the evolution of properties in the fluid surrounding the particles through integral terms. This results in systems of integro-partial differential algebraic equations (IPDAE) which may be difficult to solve through a general purpose modelling system. In fact, most of the currently available equation-oriented packages cannot even directly describe such distributed parameter systems.

The interplay of serious model developers, structuring of models for reuse and language design by experts in many engineering areas has helped to shape ICAS-MoT into its current form. Nonetheless, there are aspects of the modelling process and the development of new features in CAMS that have not yet been addressed, for example complex models, multi-scale modelling, model integration and, model documentation. Nevertheless, mathematical modelling of systems is and will continue to be a challenging activity.

The wrap-up point of this PhD project is summarized in terms of the following two issues: (a) the main achievements and contributions, and (b) the

future work in terms of remaining challenges and future directions in CAMS.

6.1 Achievements and Contributions

A generic procedure for the process of model building based on (ten) fundamental steps, that helps the developer in several modelling activities at different stages of the process life, was proposed. Based on this model building procedure, the computer-aided modelling tool ICAS-MoT was extended by developing and implementing new modelling options that aids the model developer in terms of model generation, model analysis, model translation, model solution, model validation/verification and model transfer.

The main contributions in the developed CAMS (ICAS-MoT) was the incorporation of the following new features/tools:

- Dynamic optimisation: This tool allows the solution of optimisation problems that involve DAE (Differential Algebraic Equations) and also the model parameter identification when experimental measurements (as function of time) are available.
- Matrix manipulation: This feature allows writing models in a compact notation especially when using multi-component mixtures.
- Solver for PDAE (Partial Differential Algebraic Equations): This tool allows the user to write the PDAE using an easy syntax and the equation discretisation is done in automated fashion (internally).
- Statistical report: A report is generated automatically, after the model solution, containing: (i) information about the experimental data, (ii) statistics information, such as number of data points, mean values, sum of squares, variances, standard deviations, confidential intervals, and statistics for regression, and (iii) an Anova (Analysis of variance) statistics.
- Sensitivity analysis: This option allows seeing the sensitivity response on 'explicit', 'dependent' or 'unknown' variables when some parameters (explicit or unknown variables) are perturbed.
- COM interface: This interface supports access to dynamic link libraries (DLLs) and servers based upon Component Object Model (COM). This support allows Visual Fortran (VF), Visual Basic (VB) and Visual C++ (VC++) developers to use the popular mechanisms that make functionality (services) available to other software.
- Conditional sentence: The sentence "if-then-else" was added so that a greater number of models can be implemented, analyzed and solved using ICAS-MoT.

- Process modules: Models created in ICAS-MoT can be incorporated as a module of a flowsheet in ICAS, so that custom simulators can be created using both ICASsim (steady-state simulation) and Dynsim (dynamic simulation).
- Calling procedures: the models can be decomposed into sub-models and used as external procedures (piece of code) that can be called from a main model. Nested calls are allowed.

Moreover, the strategy proposed for the model building as well as these new modelling options were applied to a wide range of application examples. Issues such as how to obtain a suitable model, how to combine process models to represent different alternatives, how to increase the application range of a model and how to specify the design targets were addressed as well. In particular, four case studies involved with various aspects of process/products design were studied and presented with enough details illustrating the full potential of ICAS-MoT. Furthermore, the modelling results provided new insights and understanding of the corresponding application fields:

1. The optimisation of an experimental anaerobic biogas process. This case study involved the development of robust solution strategies for parameter identification, dynamic optimisation and simulation. The main contributions in the area of the anaerobic biogas process were:
   - The development of a kinetic model, obtaining a trustworthy kinetic characterization validated against several experimental data.
   - The statement of a strategy of solution for the model identification of an experimental process (two-step thermophilic anaerobic digestion of primary or secondary sludge carried out in a set of two CSTRs in series), involving the solution of several sub-problems for the estimation of the unknown kinetic parameters.
   - The establishment of a modelling framework to systematically optimise the performance of the anaerobic digestion process and the identification of the main process variables, parameters and operating conditions, as well as for the efficient designing of experiments.

   In particular this case study was a key to implement and test two new important options in ICAS-MoT: Dynamic optimisation based on experimental data (used for the parameter estimation), and use of statistics report to validate and discriminate suggested kinetic models.
2. The modelling, design and operation of a polymerisation reactor. Several stages of the process lifecycle were studied, including steady state and dynamic simulations. The main contributions in the polymerisation process model were:

- The comparison of the nonlinear and linear process models for a MMA polymerisation reactor, showing the importance of including nonlinearities in the modelling of polymerisation reactors.
- The analysis of steady state (including the existence of multiplicities and its stability), open-loop analysis of the dynamic behaviour of the nonlinear and linear process model, closed-loop analysis of the dynamic behaviour under operation disturbances that allowed to understand the effect of model choice (linear or nonlinear) and the effect of the operation conditions on the estimated polymer product quality.

In this case study the use of ICAS-MoT was employed to solve and validate the open and closed-loop models, and to highlight the option of (re)use of modelling knowledge during the process lifecycle. The information incorporated in the models as well as the results were easily translated from one stage to another within the same modelling environment. Also, this case study was useful to test the option for model transfer to the ICAS environment as well as to Excel and Visual Fortran.

3. The modelling of short-path evaporation process. A complete process model was developed, tested against experimental data, and used to study various types of design/analysis problems. The main contributions in the short-evaporation process modelling were:

   - The development of a generalized model, covering a wide range of multi-component mixtures as well as operational and configurational options for the design and analysis of the purification of selected chemical products through an optimal short-path evaporator design.
   - The proposal of a systematic simulation strategy, with particular emphasis on analysis and design issues, such as industrial process operation validation, sensitivity analysis, verification/design of operational conditions for a desired separation, and, improving the yield and purity of the desired chemical product.

   The case study holds the key to implement and test the solver for PDAEs, without the solution of this process model could not be possible easily. An alterntive of solution was the manual discretisation of the PDAE system, so that a larger DAE system should have been solved, but that requires more effort to solve it.

4. The open- and closed-loop simulation of a chemical-plant. The Tennessee Eastman process model, reported in the literature as a challenge problem, was simulated reproducing the resuls earlier reported. The main contribution in this area was:

   - The implementation and simulation of both open-loop and closed-loop plant-wide model in a easy, robust and reliable way.

With this case study it was possible to illustrate the capabilities and potentials of ICAS-MoT: The computational support needed the implementation of complex and large (in terms of number of variables and equations) process models, the ability to generate customized simulators, and the successful determination of model solution in a fast, reliable and efficient manner have been highlighted.

The new features of ICAS-MoT were also tested and highlighted for a collection of process models from industry and open literature, including:

- Thermodynamic models (UNIFAC, UNIQUAC, Wilson, SRK, SAFT, PC-SAFT, CPA) –useful for chemical product design
- Kinetic models (mass action or LHHW kinetics) –useful for understanding chemical product routes
- Lumped models (mixers, chemical/polymer/bio-reactors in batch, semi-batch or continuous operation, membrane separation unit, etc.) –useful for process analysis and design
- Whole-plant models (Tennessee Eastman Plant)

The results showed that the use of ICAS-MoT for design and analysis of bio and chemical process/product are not only possible, but can be achieved in a robust, efficient and rapid manner.

6.2 Remaining Challenges and Future Direction in CAMS

Process and product design involves many disciplines and is a highly collaborative effort. One main goal in CAMS is to be able to model systems represented by any type of flowsheet, where the process units of the flowsheet are modelled at the level of accuracy dictated by the application. Flowsheet models would have unit models from different domains. One could configure planning and scheduling models that consist mostly of simplified models, or use more accurate models for process simulation, or configure plant optimisation models that use high accuracy models in critical areas of the plant. To do this, several issues must be considered for the improvement of the current CAMS:

◇ *Multi-scale modelling and integration of property models of different types within process models.* These are two topics worth investigating in future, as availability of models will lead to improved systematic methodologies for product/process design. In addition, rapid predictive cost evaluation (capital, materials and operating) could be an important CAPE challenge, as lack of systematic methodologies may mean that the evaluation process will come

too late and consequently, it could be wrongly concluded that an apparently excellent product (molecule or mixture) cannot be manufactured sustainably within the target costs. (Cordiner 2004) has discussed some of the challenges and opportunities in modelling formulations in manufacturing processes. To meet these challenges, new modelling tools as well as greater understanding of the involved phenomena will be needed ((Gani 2004)).

◇ *Model documentation.* In practice, it is very difficult to maintain models over the life cycle of a plant. On the one hand this is due to the lack of documentation as described in chapter one. On the other hand, rapidly advancing simulation technology renders the model code written in a specific representation format of some modelling environment almost useless over time. In order to prevent this problem, the knowledge associated with the model must be captured, rather than the mere code in some programming language. Ideally, the documentation should completely define all the activities and the rationale that leads to the employed model. In this case, the model can be re-engineered easily at a later time (Foss et al. 1998).

◇ *Model integration.* There is an incentive for integrating models. They could be of the same formalism but developed by means of different tools. Further, they could be even of different formalisms such as black box linear models, neural net models and first principles based models. Typically, models of different formalisms stem from different modelling environments. Integration could be on the model representation level by means of product data modelling techniques or open model representation languages, or on the procedural level where complete simulators are integrated and coordinated during run-time in the sense of heterogeneous platform simulation. Component ware could be a useful technology for systems implementation (Foss et al. 1998). But in practice, models are often only available in a variety of non-compatible formats generated by different modelling tools. Recoding these models would be tedious and expensive and not even an option if only executable code (e.g. a FORTRAN library) is available. In this case the reuse of modelling knowledge can only be achieved by actually using the model implementation as it is.

◇ *Generation of model equations.* This should be done on the basis of chemical engineering modelling knowledge, and their transformation into a system of simulation equations using the methods of applied mathematics such as required, *e.g.,* in the simulation of spatially distributed process models, is not aided by any of the present-day systems. These important modelling steps still require qualified simulation experts who posses a comprehensive knowledge of peculiar chemical engineering area, of modelling technology, and of applied mathematics (Marquardt 1994).

Summarizing, advanced CAMS are required providing means for supporting the modelling process and allowing the adaption of the modelling process to the

user needs. The development of the required concepts, the implementation in prototypical systems and finally their evaluation in an industrial environment must be viewed as a long term and demanding research agenda revealing many facets ranging from chemical engineering, software engineering and scientific computing. Only an interdisciplinary approach is expected to result in the best achievable solutions.

Finally, some suggestions for improving current modelling technology (Foss et al. 1998) are given in Table 6.1. Some of the issues have already been addressed in this thesis, however some others problems presented can only be solved in the longer run by extending and improving education in modelling and simulation as well as model-based application in the context of computer-aided modelling systems.

Table 6.1: Suggestions for improving current modelling technology

Model application	
	Sensitivity analysis
	Dynamic optimisation
	State estimation
	Control synthesis
	Mixed-integer optimisation
	Bifurcation and stability analysis
Numerical methods	
	Adaptive discretisation schemes
	Initialization
	High index problems
	Equation scaling
Model representation	
	Partial differential equations
	Integro-differential equations
	Continuous-discrete models
	Unstructured nonlinear black box modelling
	Monte Carlo modelling
	Uncertainty and disturbances, stochastic analysis
	Improve model transparency
Modelling support	
	Support for the conceptual modelling phase.
	Check physical dimensions in equations.
	Methods for systematic model reduction.
	Support the model refinement cycles.
	More advanced debugging: link numerical computations to physical model representation.
	Collection of proven dynamic process models (even if only available in written form) together with applications the models have been used in.
	Library for standardized model building blocks on a finer scale than the unit level for constructing nonstandard unit models.
	Continuous updating and improvement of libraries.
Experimental modelling	
	Experimental design
	Model structure discrimination and parameter estimation.
	Improved validation support, including the use of informal knowledge
Model reuse	
	Trace model development process for later reuse.
	Version management.
	Copy and modify.
	Abstraction of a model.
	Documentation at little extra effort, linked to model representation and simulation results, must include assumptions, rationale,quality of model, region of validity.
Results processing	
	More flexible report generation.
	Results pattern filtering.
	Visualization.
General issues	
	Better training of students in PDE modelling.
	Accelerated learning curves for proper tool usage.
	Open modelling tool (model server).
	An interface or model standard to connect models developed on different platforms.
	Heterogeneous platform simulation.
	Model life cycle aligned with process life cycle for better process and product development.

A

ICAS-MoT Operators and Functions

Table A.1: Operators list

Operators	Name	Symbol	Priority
Arithmetic			
	Unary positive, Unary negative	+x, -x	1
	Power	^	2
	Modulus	%	3
	Division	/	3
	Multiplication	*	3
	Addition, Subtraction	+, -	4
Logical			
	Less or Equal, Great or Equal	<=, >=	5
	Less Than, Greater Than	<, >	5
	Not Equal, Equal	!=, ==	5
Boolean			
	Boolean Not	!	6
	Boolean And	&&	6
	Boolean Or	||	6

Table A.2: Special operators set 1

Operator	Function
[	Array Variable
]	Array Variable
<	sum index
>	sum index
@	call procedure/function

Table A.3: Special operator set2

Operator	**sub-operator**	**Function**
partial(x,t)		partial diffeerential
	ic(z,t0)	initial condition
	lbc(z, x0)	lower boundary condition
	upc(z,x1)	upper boundary condition
if(Logical-TEST)		condition
	then (Value_if_TRUE)	accion 1
	else (Value_if_FALSE)	accion 2

Table A.4: Mathematical functions

Function	**Name**
Absolute Value / Magnitude	abs(x)
Arc Cosine	acos(x)
Hyperbolic arc cosine	acosh(x)
Arc Sine	asin(x)
Inverse hyperbolic sine	asinh(x)
Arc tangent	atan(x)
Arc tangent of a fraction y/x	atan2(y,x)
Inverse hyperbolic tangent	atanh(x)
Bessel function Im(x)	bessi(x)
Bessel function Jm(x)	bessj(x)
Bessel function Km(x)	bessk(x)
Rounded value up	ceil(x)
Trigonometric cosine	cos(x)
Hyperbolic cosine	cosh(x)
Derivative of Bessel function Im(x)	dbessi(x)
derivative of Bessel function Jm(x)	dbessj(x)
Derivative of Bessel function Km(x)	dbessk(x)
nth non-zero root of Jm(x)	djroot(x)
Exponential	exp(x)
Factorial	fact(x)
Rounded value dawn	floor(x)
nth non-zero root of Jm(x)	jroot(x)
Natural logarithm to the base e	ln(x)
Logarithm base 10	log (x)
Returns the largest number of the arguments	max(x,y)
Returns the smallest number of the arguments	min(x,y)
Returns 0 if negative or +1 if positive	post(x)
Trigonometric sine	sin(x)
Hyperbolic sine	sinh(x)
Square root	sqrt(x)
Random number (between 0 and 1)	rand()
Trigonometric tan	tan(x)
Hyperbolic tangent	tanh(x)

Table A.5: Special Functions

Function	**Name**
Summation one index(vector elements)	sum_i
Summation two indexes(matrix elements)	sum2_ij
product one index(vector elements)	mult_i

Table A.6: Thermodynamic Functions list

Thermo Functions	**Sub-functions**	**Input arguments**	**Output arguments**	**Description**
CalculateKValues		T,P,x[i],y[i]	icheckPhase, iChekFlag, CalcDerivatives KValues[i], KDerivatives[i] LiqCompressibility, VapCompressibility	Calculates K values
	GetKValues	T, P, x[i], y[i]	KValues[i]	Gets the K values from Flash calculations
	GetLiquidCompressibility	T, P, x[i]	LiqCompressibility	Calculates the liquid compressibility
	GetVaporCompressibility	T, P, y[i]	VapCompressibility	Calculates de vapour compressibility factor
CheckLiquidStability		T, P, x[i]	stable(=0), unstable(=1)	Check for liquid stability
CheckVaporStability		T, P, y[i]	stable(=0), unstable(=1)	Check for vapour stability
CalculateVaporEnthalpyF				
CalculateVaporEnthalpy		T, P, y[i]	EnthalpyVap	Calculates the vapour enthalpy
CalculateVaporHeatCapacity		T, P, y[i]	HeatCapacityVap	Calculates vapour heat capacity
CalculateLiquidEnthalpyF				
CalculateLiquidEnthalpy		T, P, x[i]	EnthalpyLiq	Calculates liquid enthalpy
CalculateLiquidHeatCapacity		T, P, y[i]	HeatCapacityLiq	Calculates the vapour enthalpy
CalculateDensity		T, P, x[i], y[i], WhichPhase	LiquidDensity, VaporDensity LiquidMolarWeigh, VaporMolarWeigh	Gets the vapour density and molar weigh
	GetLiquidMW	T, P, x[i]	LiquidMolarWeigh	Gets the liquid molar weigh
	GetVaporMW	T, P, y[i]	VaporMolarWeigh	Gets the vapour molar weigh
	GetLiquidDensity	T, P, x[i]	LiquidDensity	Get the liquid density
	GetVaporDensity	T, P, y[i]	VaporDensity	Gets the vapour density
PTFlash		T, P, z[i]	PhaseSplit, KValues[i] x[i], y[i], PhaseInfo	Flash calculation at P,T constants
VTFlash		T, Volumen, KMolarHoldup[i]	PhaseSplit, x[i], y[i]	Flash a T, V constants
BubbleP		T, x[i]	P, y[i], KValues[i]	Bubble point pressure calculation
DewP		P, y[i]	T, x[i], KValues[i]	Dew point Pressure calculation
BubbleT		P, x[i]	T, y[i], KValues[i]	Buble point temperature ccalcualtion
DewT		P, y[i]	T, x[i]	Dew point temperature calculation

Table A.7: Compound database names

MoT variable name	**Properties**
DB_Mathias_Copeman1	Mathias-Copeman first parameter
DB_Mathias_Copeman2	Mathias-Copeman second parameter
DB_Mathias_Copeman3	Mathias-Copeman third parameter
DB_AntoineA	Antoine A parameter
DB_AntoineB	Antoine B parameter
DB_AntoineC	Antoine C parameter
DB_Mw	Molecular weight [g/grmol]
DB_Omega	Acentric factor
DB_Tc	Critical Temperature [K]
DB_Pc	Critical Pressure [atm]
DB_Vc	Critical Volume [m3/kmol]
DB_Zc	Critical compressibility factor
DB_Tm	Melting point temperature [K]
DB_Tb	Boiling point temperature [K]
DB_Ttr	Ttr [K]
DB_Ptr	Ptr [atm]
DB_Vliq	Vliq [m3/kmol]
DB_igHF	igHF [kJ/kmol]
DB_igGF	igGF [kJ/kmol]
DB_igS	igS [kJ/kmolK]
DB_RG	RG [A]
DB_DM	DM [Debye]
DB_SolPar	Solubility Parameter [(MPa)0.5]
DB_VdW-Vol	van der Waals Volume [m3/kmol]
DB_VdW-Area	van der Waals Area [m2/kmol]
DB_Hfusion	Heat of fusion [kJ/kmol]
DB_Hcombust	Heat of combust[kJ/kmol]
DB_RI	RI
DB_Fpoint	Fpoint[K]
DB_Fpl	Fpl [vol%]
DB_Fpu	Fpu [vol%]
DB_AIT	AIT [K]
DB_Rborn	Born Radius
DB_DHF	Heat of formation for liquid species
DB_DHAQF	Enthalpy at standard state
DB_DGAQF	Gibbs energy at standard state
DB_DHSF	Heat of formation for solid species
DB_DGSF	Gibbs energy of formation for solid species

Table A.8: Property variable names

Properties	**A**	**B**	**C**	**D**	**E**	**F**	Equation
Solid Density [kmol/m3]	ADippr100	BDippr100	CDippr100	DDippr100	EDippr100	FDippr100	$A + B * T + C * T^2 + D * T^3 + E * T^4$
Liquid Density [kmol/m3]	ADippr101	BDippr101	CDippr101	DDippr101	EDippr101	FDippr101	$A/B^{(}1 + (1 - T/C)^D)$
Vapour Pressure [Pa]	ADippr102	BDippr102	CDippr102	DDippr102	EDippr102	FDippr102	$exp(A + B/T + C * ln(T) + D * T^E)$
Heat of Vaporization [J/kmol]	ADippr103	BDippr103	CDippr103	DDippr103	EDippr103	FDippr103	$A * (1 - Tr)^{(}B + C * Tr + D * Tr^2)$
Solid Heat Capacity [J/(kmol*K)]	ADippr104	BDippr104	CDippr104	DDippr104	EDippr104	FDippr104	$A + B * T + C * T^2 + D * T^3 + E * T^4$
Liquid Heat Capacity [J/(kmol*K)]	ADippr105	BDippr105	CDippr105	DDippr105	EDippr105	FDippr105	$A + B * T + C * T^2 + D * T^3 + E * T^4$
Ideal Gas Heat Capacity [J/(kmol*K)]	ADippr106	BDippr106	CDippr106	DDippr106	EDippr106	FDippr106	$A + B * (C/T/sinh(C/T))^2 + D * (E/T/cosh(E/T))^2$
Second Virial Coefficient [m3/kmol]	ADippr107	BDippr107	CDippr107	DDippr107	EDippr107	FDippr107	$A + B/T + C/T^3 + D/T^8 + E/T^9$
Liquid Viscosity [kg/(m*s)]	ADippr108	BDippr108	CDippr108	DDippr108	EDippr108	FDippr108	$exp(A + B/T + C * ln(T) + D * T^E)$
Vapour Viscosity [kg/(m*s)]	ADippr109	BDippr109	CDippr109	DDippr109	EDippr109	FDippr109	$A * T^B/(1 + C/T + D/T^2)$
Liquid Thermal Conductivity [J/(m*s*K)]	ADippr110	BDippr110	CDippr110	DDippr110	EDippr110	FDippr1110	$A + B * T + C * T^2 + D * T^3 + E * T^4$
Vapour Thermal Conductivity [J/(m*s*K)]	ADippr111	BDippr111	CDippr111	DDippr111	EDippr111	FDippr111	$A * T^B/(1 + C/T + D/T^2)$
Surface Tension [kg/s2]	ADippr112	BDippr112	CDippr112	DDippr112	EDippr112	FDippr112	$A * (1 - Tr)^{(}B + C * Tr + D * Tr^2)$

B

Parameter Estimation

B.1 Parameter Estimation

Parameter estimation arises in many different areas of engineering, where mathematical models are used to describe real life phenomena and experiments are performed to validate these models. Advantages of mathematical models include the ability to do optimisation of design and production and the ability to analyse and understand system behaviour subject to conditions that are not readily handled by experiments. Often the models contain a number of parameters that cannot be measured directly or calculated by applying established laws of nature, and therefore must be estimated from experimental data. The basic concept is to determine these parameters such that the differences between the experimental data and the values predicted by the model are minimal in some sense: The predicted, theoretical values should fit the measurements. The choice of fitting criterion depends on the knowledge and the assumptions about the measurement errors. This section addresses the problem of estimating parameters in dynamical models, especially those described by ordinary differential equations (ODEs) or differential algebraic equations (DAEs). Methods tailored for partial differential equation (PDE) models are not discussed here, but often these models can be reduced to a set of ODEs, which allows the use of techniques developed for ODEs. Estimating parameters in dynamical models is computationally intensive, since it requires the repeated (numerical) solution of the underlying set of differential equations. Efficient and robust methods for solving this problem are important for the development and improvement of process models (Kristensen (2004)).

- Least squares. One of the most widely used methods of estimation is *least squares* (LS). In its simplest form the parameters are estimated such that the sum of squared residuals:

$$f(\theta) = \frac{1}{2}\sum_{i=1}^{m} r_i^2(\theta) \tag{B.1}$$

  is minimal. The function $f : \mathbb{R}^{n_p} \mapsto \mathbb{R}$ denotes the least squares *objective function*. The factor $\frac{1}{2}$ is introduced for convenience to avoid a factor

of 2 when taking the derivative of f. The least squares criterion can be regarded as a maximum *likelihood* (ML) criterion, if certain assumptions about the distribution of measurement errors are made. This shows that, provided these assumptions hold, the least squares estimates possess optimal statistical properties. The fitting criterion (B.1) is also referred to as ordinary least squares (OLS). This criterion is often unsatisfactory for the following reasons (Bard (1974)):

- The measured quantities may have different physical dimensions, or may be measured on different scales. For example, some of the measurements in the measurement vector $\tilde{\mathbf{y}}$ may represent concentrations of a chemical species, expressed in mole fractions and falling into the range of zero to one. Other measurements, however, may be temperature measured in Kelvin with values in a much higher range. Fitting an OLS criterion will most likely result in the temperature residuals dominating those of the mole fractions, and any information contained in the latter will be lost.
- Some measurements may be known to be less reliable than others. Thus, a tool is needed to make sure that the parameter estimates are less influenced by these measurements relative to the more accurate ones.

The solution to both of these problems is to introduce weight factors into the objective function resulting in a *weighted least squares* (WLS) criterion:

$$f(\theta) = \frac{1}{2} \sum_{i=1}^{m} w_i^2 r_i^2(\theta) \tag{B.2}$$

The weights are chosen small, if the measurements are unreliable or measured on a large scale and large if the opposite is true. An implicit assumption of OLS and WLS is that the errors are only present in the dependent variables. That is, the independent variable (often time) is assumed to be known without error. An approach considering errors in all variables is *total least squares* (TLS), also referred to as orthogonal distance regression (Stortelder (1998)). If the measurement errors in time are denoted $\varepsilon_i \, (i = 1, \ldots, m)$, the measured time is given by:

$$\tilde{t}_i = t_i + \varepsilon_i \tag{B.3}$$

The residuals related to the independent variables are denoted δ_i. Thus, the overall residuals between the measurements and the theoretical values now depend on both θ and δ:

$$r_i(\theta, \delta_i) = y_{ci}(t_i + \delta_i, \theta) - \tilde{y}_i \tag{B.4}$$

This leads to the following TLS fitting criterion:

$$f(\theta, \delta) = \frac{1}{2} \sum_{i=1}^{m} w_i^2 r_i^2 (\theta, \delta_i) + d_i^2 \delta_i^2 \tag{B.5}$$

in which d_i denotes the weight associated with the ith residual of the independent variable. If the weights in (B.5) are chosen equal, then minimization of the TLS criterion corresponds to minimizing the orthogonal distance between the measurements and the curve $y(t, \theta)$, hence the name orthogonal distance regression.

- Maximum likelihood

  The maximum likelihood estimates are derived from the probability density function of the measurement errors. Under certain assumptions these estimates coincide with the OLS or WLS estimates, which will be discussed in the following. In this section the errors are assumed to be present only in the dependent variables. The measurement errors ε_i are assumed mutually independent and normally distributed with zero mean and variance σ^2, i.e. $\varepsilon_i \sim \mathcal{N}(\theta, \sigma^2)$. The covariance matrix for the vector of measurement errors is:

$$V = E\left\{\varepsilon \varepsilon^T\right\} = \sigma^2 I_m \tag{B.6}$$

  where E denotes the expectation operator.

  If the residuals are assumed to give an adequate representation of the measurement errors, the probability density function for the assumed structure of the measurement errors is given by (Seber and Wild (2003)):

$$\begin{aligned} p(\tilde{y}|\,\theta) &= \left(2\pi\sigma^2\right)^{-m/2} \exp\left(\frac{-\sum_{i=1}^{m} r_i^2(\theta)}{2\sigma^2}\right) \\ &= \left(2\pi\sigma^2\right)^{-m/2} \exp\left(\frac{1}{2} r(\theta)^T V^{-1} r(\theta)\right) \end{aligned} \tag{B.7}$$

  in which $\mathbf{r} : \mathbb{R}^{\mathbf{n_P}} \mapsto \mathbb{R}^{\mathbf{m}}$ denotes a vector of assembled residuals. The maximum likelihood estimate is the value of θ that maximizes the probability density function, i.e. the most likely θ for a given data set. The likelihood function is defined as:

$$L(\theta) = p(\tilde{y}|\,\theta) \tag{B.8}$$

  Taking the logarithm of the likelihood function yields:

$$\ln L(\theta) = -\frac{m}{2}\ln\left(2\pi\sigma^2\right) - \frac{1}{2}r(\theta)^T V^{-1} r(\theta) \tag{B.9}$$

The likelihood function reaches its maximum when the latter term in (B.9) is minimal, which corresponds to the minimum of the OLS objective function (B.1). Thus, in the case of independent and identically distributed measurement errors from a normal distribution, the maximum likelihood estimate of θ coincides with the OLS estimate. A connection between maximum likelihood and WLS also exists. If the measurement errors are assumed independent and normally distributed, but with non-constant variances, $\varepsilon_i \sim \mathcal{N}\left(\theta, \sigma^2\right)$, the corresponding likelihood function is (Seber and Wild (2003)):

in which V now has non-constant diagonal elements $V_{ii} = \sigma_i^2$. Again, taking the logarithm yields:

$$L(\theta) = -\frac{(2\pi)^{-m/2}}{\sqrt{\det(V)}}\exp\left(-\frac{1}{2}r(\theta)^T V^{-1} r(\theta)\right) \tag{B.10}$$

Comparing (B.2) and (B.10) shows that the maximum likelihood estimates coincide with the WLS estimates, if the weights in (B.2) are chosen proportional to the reciprocal of the standard deviations:

$$w_i \propto \frac{1}{\sigma_i} \tag{B.11}$$

This establishes the connection between maximum likelihood and WLS. However, in most practical situations the standard deviations of the measurement errors are unknown. The standard deviations can be estimated along with the unknown parameters. If the measurement errors are assumed independent, the covariance matrix is diagonal. In the more general case, the covariance matrix is a full matrix. To limit the number of unknown elements to be estimated, assumptions such as independence of errors at different points in time and zero expectation of errors are often made. Details on maximum likelihood estimation with unknown covariance can be found in Bard (1974). These cases may be regarded as WLS problems with unknown weights. As mentioned earlier, the choice of estimation method depends on the assumptions and knowledge about the measurement errors. The conditions required to apply a maximum likelihood criterion are often not present, e.g. an incorrect model structure results in systematic errors. On the other hand, a least squares criterion may lack a sound statistical interpretation, unless assumptions similar to maximum likelihood with known variances are made, in which case the two types of estimation coincide.

B.2 Statistical Analysis

Here, a brief discussion is given on estimation of the covariance and calculation of confidence regions for the parameter estimates. In this section the errors in the measurements are assumed independent and normally distributed with zero expectation and known variance σ^2. It is important to obtain an estimate of the correlation between the estimated parameters. Correlation between parameters indicates that insufficient information is available in the data to estimate the model parameters uniquely, so that either the model structure should be reconsidered or further experiments performed. If $\hat{\theta}$ denotes the estimated parameter vector, then the covariance matrix for the parameters can be approximated as follows (Seber and Wild (2003)):

$$\operatorname{cov}\left(\hat{\theta}\right) = E\left\{\left(\theta * -\hat{\theta}\right)\left(\theta * -\hat{\theta}\right)^T\right\} \approx \hat{\sigma}^2\left(H\left(\hat{\theta}\right)\right)^{-1} \tag{B.12}$$

in which $H\left(\hat{\theta}\right)$ denotes the Hessian matrix of the objective function evaluated at $\hat{\theta}$. $\hat{\sigma}^2$ is an estimator for σ^2:

$$\hat{\sigma}^2 = \frac{f\left(\theta\right)}{m - n_p} \tag{B.13}$$

The covariance estimate B.12 can be derived from a linearisation of the objective function around θ^*. An often used approximation to the Hessian matrix is the so-called Gauss-Newton approximation allows to rewritten B.12 as:

$$\operatorname{cov}\left(\hat{\theta}\right) = \approx \hat{\sigma}^2\left(\mathbf{J}\left(\hat{\theta}\right)^T\mathbf{J}\left(\hat{\theta}\right)\right)^{-1} \tag{B.14}$$

Details on the properties of the covariance estimate are found in Seber and Wild (2003). The only other statistical element considered here is an approximate confidence interval for the parameter estimates. Assuming that the estimator for the parameters is normally distributed:

$$\hat{\theta} \sim N\left(\theta *, \sigma^2 C\right), \qquad C = \left(H\left(\hat{\theta}\right)\right)^{-1} \tag{B.15}$$

a 100% marginal confidence interval for the i^{th} parameter is given by:

$$\hat{\theta}_i \mp t_{m-n_p}^{\alpha/2}\hat{\sigma}^2\sqrt{C_{ii}} \tag{B.16}$$

in which $t_{m-n_p}^{\alpha/2}$ denotes a quantile of the t-distribution with $m - n_p$ degrees of freedom. In the case of high correlation between the parameters the marginal intervals might be misleading, since the correlation is not taken into account. Approaches exist for constructing simultaneous intervals (Seber and Wild (2003)).

C

PDEs Classification and its Solution Procedures

Table C.1: PDE Classification

Orden in z(BC)	**Order in t(IC)**	**type**	**Example**
1	1	First order hyperbolic	$\frac{\partial u}{\partial t} = -v\frac{\partial u}{\partial z}$ (advection equation)
2	2	Second order hyperbolic	$\frac{\partial^2 u}{\partial t^2} = -c^2\frac{\partial^2 u}{\partial z^2}$ (wave equation)
2	1	Parabolic	$\frac{\partial u}{\partial t} = \alpha\frac{\partial^2 u}{\partial z^2}$ (Fourier or Fick's 2nd law)
2 (in y, z)	0	Elliptic	$\frac{\partial^2 u}{\partial z^2} + \frac{\partial^2 u}{\partial y^2} = 0$ (Laplace's equation)

C.1 Solving Partial Differential Equation

Dynamic simulation plays an important role in analysing and predicting the time transient behaviour of chemical and biochemical processes and is therefore widely used, especially in the last decade. But, as pointed out long time ago by Heydweiller, Sincovec and Fan (1977) and confirmed more recently in the interesting work of Oh (1995), most existing dynamic modelling and simulation tools are primarily suited to lumped parameter systems. In fact, the reality, a large number of unit operations in Chemical Engineering such as catalytic reactors, absorption column, extraction column based on mass transfer approach and in biochemical engineering are intrinsically distributed in nature. It means that their properties exhibit spatial as well as temporal variations. The resulting set of equations for these types of models may be viewed as a combination of lumped and distributed parameter systems, namely described by Partial Differential Equations (PDAE) submitted to initial conditions and boundary conditions. Partial Differential equations (PDE) and, by the way, PDAEs arise in an enormous number of modelling applications. They are difficult to solve because a change in a parameter or one of the boundary condition may lead to completely different behaviour. So, although numerical methods are suitable to accurately solve a given PDE system, other numerical methods may be totally unable to do so. When extending that to PDAEs, it is becoming more and more difficult and may completely stress the numerical method. For non specialists and even for specialists solving, PDAEs look like an inextricable numerical jungle. Because of this, it appears there is a lack of universally applicable solution methods in spite of the current effort in the field.

Among the large number of numerical methods devoted to the solution of PDE or PDAE systems which are by nature very difficult to solve, a well established classification would be the following ones.

Numerical methods may be based on :

- the use of Method of lines (MOL) (Carver (1981); Schiesser (1996); Schiesser (1991))
- Finite difference Methods (Anderson, Tannehill and Pletcher (1992))
- Weighted Residuals Methods (Finlayson (1980); Villadsen and Stewart (1967))
- Finite Element Methods (Strang and Fox (1973); Carey and Finlayson (1975))
- Finite Volume Methods (Pantakar (1980))
- Adaptive Grid Methods (Sanz-Serna and Verwer (1989); Arney and Flaherty (1990))
- Moving Grid Methods (Miller (1981))

As mentioned previously, all of them are of interest, with pros and cons from a numerical point of view. MOL, basically convert PDEs into sets of DOE with respect to time by involving space discretisation. The main advantage results in that sophisticated framework based on well established methods [LSODI, (Byrne, Hindmarsh, Jackson and Brown (1977)); DASSL (Petzold (1982)), DASOLV (Jarvis and Pantelides (1991)), RESEDA (Le-Lann and Sargousse (1996))] may be used for large set of ODEs or DAEs. But as a main drawback, general ODE or DAE solvers have difficulties to control and estimate the impact of the space discretisation error on the general numerical scheme, especially when coarse spatial grids have to be used. The most interesting feature of the finite volume formulation is that the resulting solution ensures that the conservation of quantities involved such as mass, momentum and energy is exactly satisfied not only over any group of control volumes but over the whole computation domain, which is not the reality when dealing with finite difference methods.

Finite element approach takes advantage in that ability to divide domain of interest in elementary sub-domains, namely elements. It may therefore handle problems with steep gradients and may deal with irregular geometric configurations.

Adaptive and moving grid methods seem to be the most promising in the fact that the idea is to use a numerical method in which nodes are automatically positioned in order to follow or anticipate deep fronts. It may be done by using two basic strategies, namely the spatial redistribution of a fixed number of points and the local grid refinement. The principal advantage of this method is that the original grid structure is preserved, but at the detriment of computer time and storage increase. Furthermore, as mentioned in (Brenan et al. 1989), coding of such techniques is relatively difficult and may throw complex problems as discontinuities, frequent restarting of the numerical methods.

When looking to the most recent literature and as mentioned by Oh (1995) in its attempts to propose a general framework under gPROMS (Pantelides and Barton (1994); Oh and Pantelides (1995)) dealing with modelling and simulation of combined lumped and distributed processes, the aim of constructing reliable software capable of solving a wide spectrum of PDE/PDAE/IPDAE problems has not yet fulfilled and even does not look reachable in the nearest future. Some reasons for that (Machura and Sweet (1980); Oh and Pantelides (1995))

- PDE problems allow a considerable freedom of formulation
- Some specific features of PDE problems require very specific treatment
- There exist so many numerical methods designed to deal with small classes of problems, none of them being truly universal.

Despite this pessimistic view, constant effort have been developed into two main directions :

Domain-specific packages designed to deal with a particular physical problem domain such as CFD (Computational Fluid Dynamics) with typical CFD package such as PHOENICS (CHAM (1987)), based on finite volume techniques, FLUENT, ESTET, POLY3D and General-purpose packages. such as PDECOL (Madsen and Sincovec (1980)), DSS/2 (Schiesser (1991)). High-level PDE packages such as PDEDIS (Pfeiffer and Marquardt (1996)) have to be mentioned also in this context.

In conclusion of this part, which has been particularly devoted to the PDEs background, it may be notice that problems of interest to process engineering are usually described as PDAE systems, even as IPDAE systems when dealing with population models (Ramkrishna (1985); Toutain, Joulia, Gourdon and LeLann (1998)). More, these dynamic models try to be the most "high fidelity" models used and applied in realistic operational environments. It means that they may be used in various configurations such as abrupt changes, start-up and shut down configurations

These models should further be used for simulation, including parameter estimation, even in the near future for fast real time computing. It may involve in realistic environment the necessity of state events detection, treatment of internal discontinuities (change in model mode, for example, single phase to two or three phase systems) and external discontinuities due to deep change on feed or on operation parameter.

According to that, the most pragmatic approach seems to be that belonging to the MOL approach with two stages as suggested by Oh (1995). A rough sketch of it is presented on figure C.1. At first, PDAEs are discretised in terms of finite dimensional representations, using the discretisation schemes (finite differences or finite volume approach in order to insure balances conservation). By the way, PDAEs are transformed into DAEs with respect to time. Then, DAEs are integrated using appropriate integration techniques.

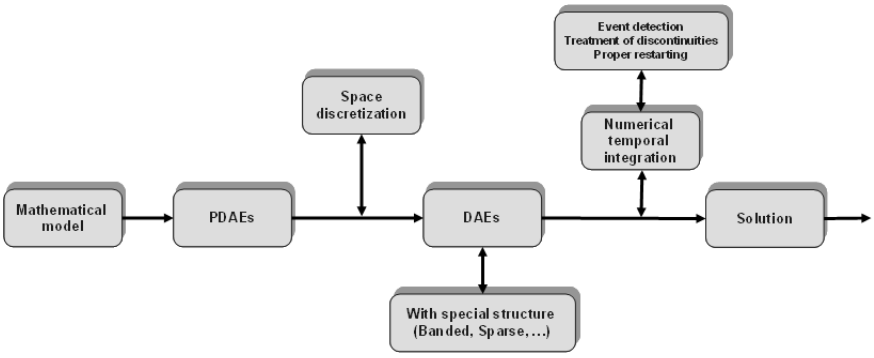


Figure C.1: Numerical strategies for PDAEs solution.

C.1.1 Decision Tree to solve PDAE systems

The classification given by Table C.1 is useful for: (a) the definition of the PDE type and (b) the suggestion of numerical methods for the solution. Both of these issues generate the following decision trees to solve PDE systems.

First, in the main Tree 1 (Figure C.2, a PDE system is input and classified as elliptic, hyperbolic, parabolic or other special types. Afterwards, depending on the type, decision Trees 2 (Figure C.3 to Figure C.6 are followed. Some details of the specific procedures (1 to 22) are given [Numerical Algorithm Group (NAG). http://www.nag.co.uk/.]

C.1.2 Procedures for first order systems

1. Procedure 1. It integrates a system of linear or non-linear, first-order, time-dependent in one space variable. The method of lines is employed to reduce the system into a set of ordinary differential equations (ODEs). The resulting system is solved using a Backward Differentiation Formula (BDF) method.

2. Procedure 2. It integrates a system of linear or non-linear, first-order, time-dependent in one space variable, with scope for coupled ODEs. The method of lines is employed to reduce the PDAEs to a system of ODEs. The resulting system is solved using BDF method or a Theta method (switching between Newton's method and functional iteration).

3. Procedure 3. It integrates a system of linear or non-linear, first-order, time-dependent in one space variable, with scope for coupled ODEs, and automatic adaptive spatial re-meshing. The method of lines is employed to reduce the PDAEs to a system of ODEs. The resulting system is solved using a BDF method or a Theta method.

C.1.3 Procedures for elliptic systems

1. Procedure 4. It solves Laplace's equation in two dimensions for an arbitrary domain bounded internally or externally by one or more closed contours, given the value of either the unknown function or its normal derivative (into the domain) at each point of the boundary.

2. Procedure 5. It uses the Strongly Implicit Procedure to calculate the solution to a system of simultaneous algebraic equations of five-point molecule form on a two-dimensional topologically-rectangular mesh (meaning that a polar grid.

3. Procedure 6. It uses the Strongly Implicit Procedure to calculate the solution to a system of simultaneous algebraic equations of seven-point molecule form on a three-dimensional topologically-rectangular mesh.

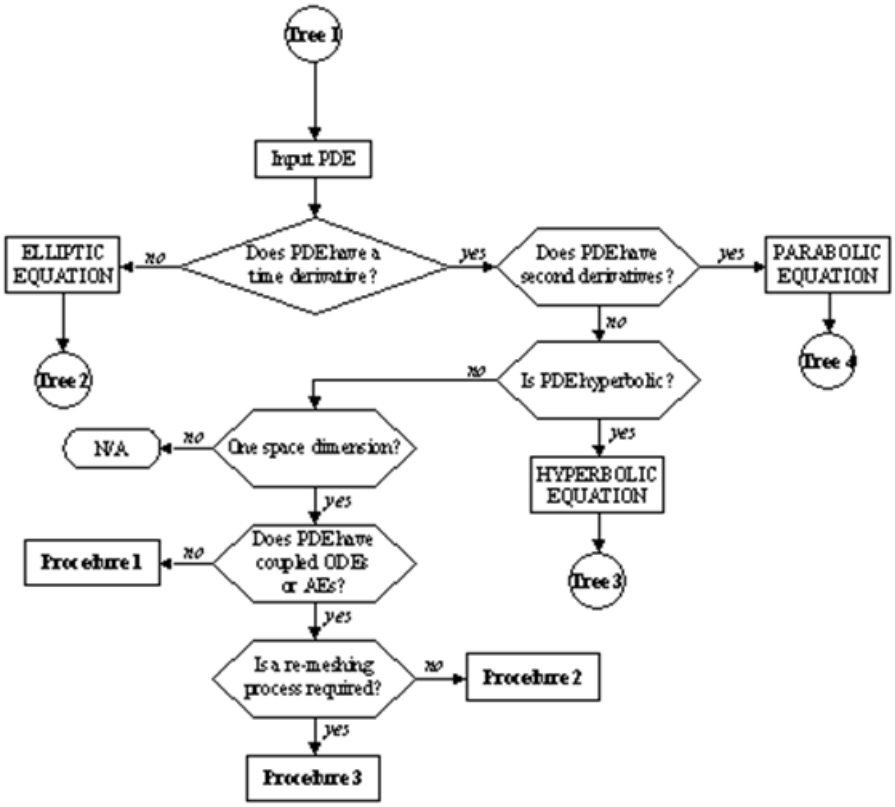


Figure C.2: Decision Tree 1: Solution of a PDAE system

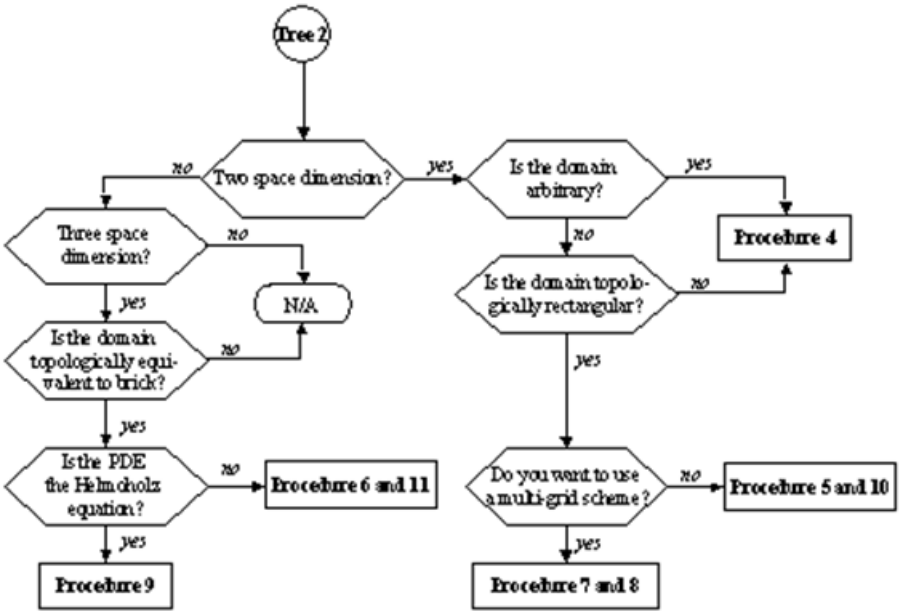


Figure C.3: Decision Tree 2: Elliptic Branch

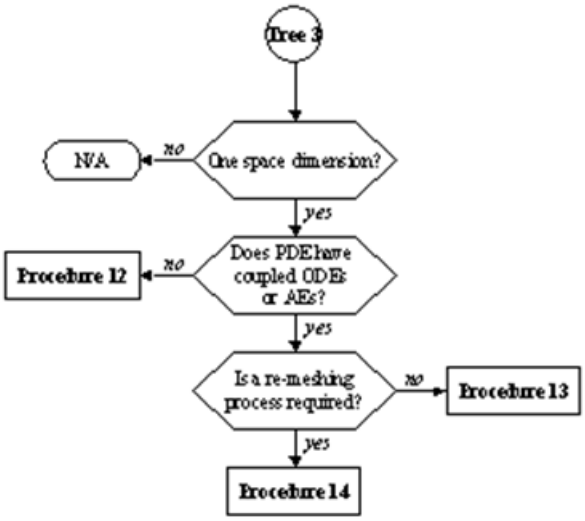


Figure C.4: Decision Tree 3: Hyperbolic Branch

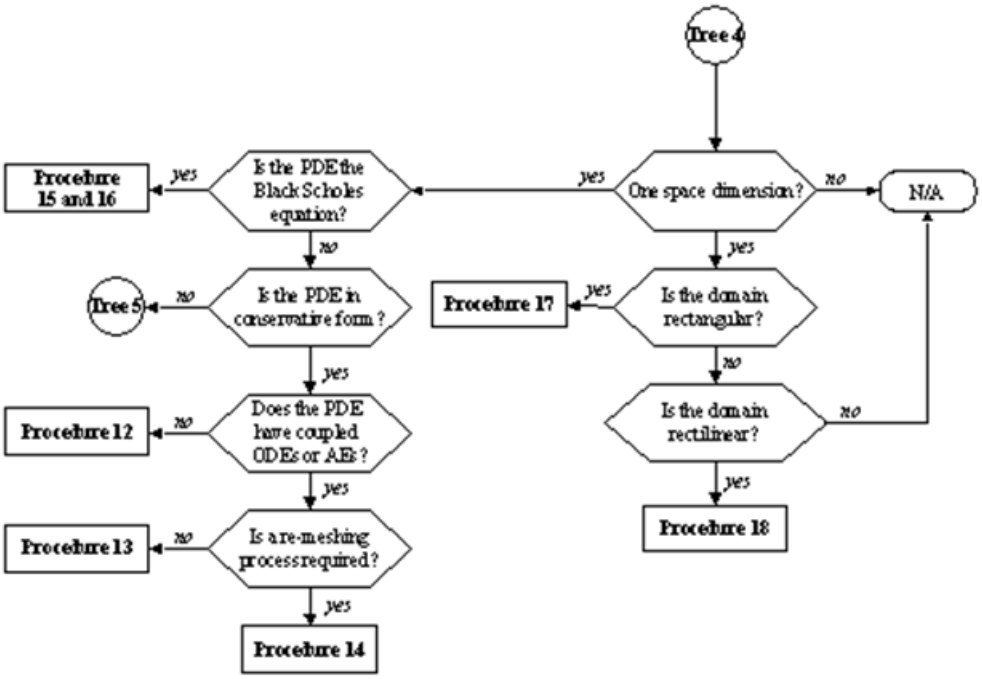


Figure C.5: Figure 4. Decision Tree 4: Parabolic Branch

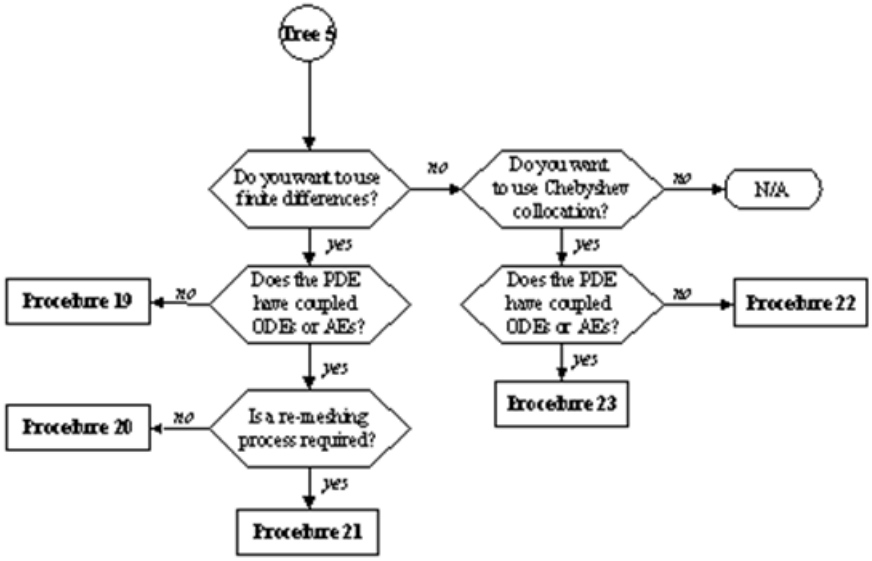


Figure C.6: Decision Tree 5: Parabolic Branch (PDAE in non-conservative form)

4. Procedure 7. It solves seven-diagonal systems of linear equations that arise from the discretisation of an elliptic PDE on a rectangular region, using a multi-grid technique.

5. Procedure 8. It discretises a second-order elliptic PDE on a rectangular region.

6. Procedure 9. It solves the Helmholtz equation in Cartesian co-ordinates in three dimensions using the standard seven-point finite difference approximation.

C.1.4 Basic SIP (Strongly Implicit Procedures)

1. Procedure 10. It performs at each call one iteration of the Strongly Implicit Procedure. It is used to calculate on successive calls a sequence of approximate corrections to the current estimate of the solution when solving a system of simultaneous algebraic equations for which the iterative up-date matrix is of five-point molecule form on a two-dimensional topologically-rectangular mesh.

2. Procedure 11. It performs at each call one iteration of the Strongly Implicit Procedure. It is used to calculate on successive calls the sequence of approximate corrections to the current estimate of the solution when solving a system of simultaneous algebraic equations for which the iterative up-date matrix is of seven-point molecule form on a three-dimensional topologically-rectangular mesh.

C.1.5 Procedures for convection-diffusion systems

1. Procedure 12. It integrates a system of linear or non-linear convection-diffusion equations in one space dimension, with optional source terms. The system must be posed in conservative form. Convection terms are discretised using a sophisticated upwind scheme involving a user-supplied numerical flux function based on the solution of a Riemann problem at each mesh point. The method of lines is employed to reduce the PDEs to a system of ODEs, and the resulting system is solved using a BDF method.

2. Procedure 13. It integrates a system of linear or non-linear convection-diffusion equations in one space dimension, with optional source terms and scope for coupled ODEs. The system must be posed in conservative form. Convection terms are discretised using a sophisticated upwind scheme involving a user-supplied numerical flux function based on the solution of a Riemann problem at each mesh point. The method of lines is employed to reduce the PDEs to a system of ODEs, and the resulting system is solved using a BDF method or a Theta method.

3. Procedure 14. It integrates a system of linear or non-linear convection-diffusion equations in one space dimension, with optional source terms and scope for coupled ODEs. The system must be posed in conservative form. This routine also includes the option of automatic adaptive spatial re-meshing. Convection terms are discretised using a sophisticated upwind scheme involving a user supplied numerical flux function based on the solution of a Riemann problem at each mesh point. The method of lines is employed to reduce the PDEs to a system of ODEs, and the resulting system is solved using a BDF method or a Theta method.

C.1.6 Procedures for Black-Scholes equation

1. Procedure 15. It solves a Black-Scholes equation using finite-difference scheme.

2. Procedure 16. It computes an analytical solution of a Black-Scholes equation for a certain set of option types.

3. Procedures for second order systems.

4. Procedure 17. It integrates a system of linear or non-linear, time-dependent PDEs in two space dimensions on a rectangular domain. The method of lines is employed to reduce the PDEs to a system of ODEs, which are solved using a BDF method. The resulting system of non-linear equations is solved using a modified Newton method.

5. Procedure 18. It integrates a system of linear or non-linear, time-dependent PDEs in two space dimensions on a rectilinear domain. The method of lines is employed to reduce the PDEs to a system of ODEs, which are solved using a BDF method. The resulting system of non-linear equations is solved using a modified Newton.

C.1.7 Procedures for parabolic systems

1. Procedure 19. It integrates a system of linear or non-linear parabolic PDEs in one space variable. The spatial discretisation is performed using finite differences, and the method of lines is employed to reduce the PDEs to a system of ODEs. The resulting system is solved using a BDF method.

2. Procedure 20. It integrates a system of linear or non-linear parabolic PDEs in one space variable with scope for coupled ODEs. The spatial discretisation is performed using a finite differences, and the method of lines is employed to reduce the PDEs to a system of ODEs. The resulting system is solved using a BDF or a Theta method (switching between Newton's method and functional iteration).

3. Procedure 21. It integrates a system of linear or non-linear parabolic PDEs in one space variable, with scope for coupled ODEs, and automatic adaptive spatial re-meshing. The spatial discretisation is performed using a finite differences, and the method of lines is employed to reduce the PDEs to a system of ODEs. The resulting system is solved using a BDF or a Theta method (switching between Newton's method and functional iteration).

4. Procedure 22. It integrates a system of linear or non-linear parabolic PDEs in one space variable. The spatial discretisation is performed using a collocation method, and the method of lines is employed to reduce the PDEs to a system of ODEs. The resulting system is solved using a BDF method.

5. Procedure 23. It integrates a system of linear or non-linear parabolic PDEs in one space variable with scope for coupled ODEs. The spatial discretisation is performed using a collocation method, and the method of lines is employed to reduce the PDEs to a system of ODEs. The resulting system is solved using a BDF or a Theta method (switching between Newton's method and functional iteration).

D

ICAS-MOT examples source code

D.1 Short examples

D.1.1 PC-SAFT EOS

```
# PC-SAFT
# *********
# Compressibility factor calculation using PC-SAFT EOS Model
#************************
#*MSC                    *
#*CAPEC, DTU, DK         *
#*13.01.05               *
#************************
#
#Variable description
#*
#* Psys          Pressure. [Pa]
#* T             Temperature, [K]
#* x[i]          mole fraction of chains of component i

#* a{n}          dispersion term parameters for chain molecules (n=0...6)
#* b{n}          dispersion term parameters for chain molecules (n=0...6)
#* pa{n}_x[k]    partial derivative of a{n} with respect to mole fraction
n=0...6)
#* pb{n}_x[k]    partial derivative of b{n} with respect to mole fraction
#* pahsxk[i]     partial derivative of ahs with respect to mole fraction

#* zeta{n}       abbreviation (n=0..3)

#* d[i]          Temperature-dependent segment diameter,[A]
#* g[i,j]hs      Radial pair distribution function for segments of
                 component i in the hasr
sphere system.
```

```
#* I1, I2        abbreviations
#* k[i,j]        segement interactions of unlike chains (binary
                 interaction parameters)
#* m[i]          number of segments in a chain of component i
#* mm            mean segment number in the system
#* rho           the total number density of molecules, [1/A3]
#* rhom          molar desity, [kmol/m3]
#* Z             compressibility factor
#* Zhc           Residual contribution of hard-chain system
#* Zhs           Residual contribution of hard-sphere system
#* Zpert         Perturbation contribution

#* adisp         dispersion contribution to the Helmholtz free energy
#* A1kTN         first order contribution to dispersion contribution
#* A2kTN         second order contribution to dispersion contribution
#* ares          residual Helmholtz free energy
#* ahc           Residual contribution of hard-chain system
#* ahs           residual contribution of the hard-sphere fluid to the
                 Helmholtz free energy

#* pahsxk[i]    partial derivative of ahs with respect to mole fraction
#* pahcxk[i]    partial derivative of ahc with respect to mole fraction
#* padispxk[i]  partial derivative of adisp with respect to mole fraction
#* paresxk[i]   partial derivative of adisp with respect to mole fraction


#* mureskT[i]    residual chemical potential
#* phi[i]        Fugacity coefficient


#* eta           packing fraction
#* epsilon       depth of pair potential, [J]
#* sigma         segment diameter, [A]
#* zeta[n]       abbreviation (n = 0..3), [An-3]

#* a01, a02, a03 model constant
#* b01, b02, b03 model constant

#* RGAS          Universal gas constant, [J/(mol K)]
#* NAV           Avogadro's number, [molecules/mole ]
#* kBOLTZ        Boltzamann constant, [J/K] = RGAS/NAV
#* NAV           Avogadro's number, [molecules/mole ]
#* PI            PI number = acos(-1)

#* The mixture Butane-Methane is considered as a test system.*
```

```
#The model

#segment diameter of component i
d[i]  = sigma[i]*(1 - 0.12*exp(-3*epsilonk[i]/T))

#segment diameter
dd{0}[i] = d_0*d[i]/(d_0 + d[i])
dd{1}[i] = d_1*d[i]/(d_1 + d[i])

#combining rules for a pair of unlike segments
sigma{0}[i] = 0.5*(sigma_0  + sigma[i])
sigma{1}[i] = 0.5*(sigma_1  + sigma[i])

epsilonk{0}[i] = sqrt(epsilonk_0*epsilonk[i])*(1 - k{0}[i])
epsilonk{1}[i] = sqrt(epsilonk_1*epsilonk[i])*(1 - k{1}[i])

#mean segment number in the mixture
mm = sum_i(x[i]*m[i])

#abbreviation for the first-order perturbation term
sum{j}_0= sum_i(x_0*x[i]*m_0*m[i]*(epsilonk{0}[i]/T)*sigma{0}[i]^3)
sum{j}_1= sum_i(x_1*x[i]*m_1*m[i]*(epsilonk{1}[i]/T)*sigma{1}[i]^3)

m2es3_m =sum_i(sum{j}[i])

#abbreviation for the second-order perturbation term
sum{j}_0= sum_i(x_0*x[i]*m_0*m[i]*(epsilonk{0}[i]/T)^2*sigma{0}[i]^3)
sum{j}_1= sum_i(x_1*x[i]*m_1*m[i]*(epsilonk{1}[i]/T)^2*sigma{1}[i]^3)

m2e2s3_m =sum_i(sum{j}[i])

#power series coefficient of first-order dispersion term for chain molecules
a{0} = a{0}{0} + (mm - 1)/mm*a{1}{0} + (mm - 1)/mm*(mm - 2)/mm*a{2}{0}
a{1} = a{0}{1} + (mm - 1)/mm*a{1}{1} + (mm - 1)/mm*(mm - 2)/mm*a{2}{1}
a{2} = a{0}{2} + (mm - 1)/mm*a{1}{2} + (mm - 1)/mm*(mm - 2)/mm*a{2}{2}
a{3} = a{0}{3} + (mm - 1)/mm*a{1}{3} + (mm - 1)/mm*(mm - 2)/mm*a{2}{3}
a{4} = a{0}{4} + (mm - 1)/mm*a{1}{4} + (mm - 1)/mm*(mm - 2)/mm*a{2}{4}
a{5} = a{0}{5} + (mm - 1)/mm*a{1}{5} + (mm - 1)/mm*(mm - 2)/mm*a{2}{5}
a{6} = a{0}{6} + (mm - 1)/mm*a{1}{6} + (mm - 1)/mm*(mm - 2)/mm*a{2}{6}

#power series coefficient of second-order dispersion term for chain molecules
b{0} = b{0}{0} + (mm - 1)/mm*b{1}{0} + (mm - 1)/mm*(mm - 2)/mm*b{2}{0}
b{1} = b{0}{1} + (mm - 1)/mm*b{1}{1} + (mm - 1)/mm*(mm - 2)/mm*b{2}{1}
```

```
b{2} = b{0}{2} + (mm - 1)/mm*b{1}{2} + (mm - 1)/mm*(mm - 2)/mm*b{2}{2}
b{3} = b{0}{3} + (mm - 1)/mm*b{1}{3} + (mm - 1)/mm*(mm - 2)/mm*b{2}{3}
b{4} = b{0}{4} + (mm - 1)/mm*b{1}{4} + (mm - 1)/mm*(mm - 2)/mm*b{2}{4}
b{5} = b{0}{5} + (mm - 1)/mm*b{1}{5} + (mm - 1)/mm*(mm - 2)/mm*b{2}{5}
b{6} = b{0}{6} + (mm - 1)/mm*b{1}{6} + (mm - 1)/mm*(mm - 2)/mm*b{2}{6}

#representation of perturbation integral of first and second order respectively
I1 = a{0}*eta^0 + a{1}*eta^1 + a{2}*eta^2 + a{3}*eta^3 + a{4}*eta^4 +
 a{5}*eta^5 +
 a{6}*eta^6
I2 = b{0}*eta^0 + b{1}*eta^1 + b{2}*eta^2 + b{3}*eta^3 + b{4}*eta^4 +
 b{5}*eta^5 +
 b{6}*eta^6

C1 =( 1 + mm*(8*eta - 2*eta^2)/(1 - eta)^4 + (1 - mm)*(20*eta - 27*eta^2 +
 12*eta^3 -
 2*eta^4)/((1 - eta)*(2 - eta))^2)^(-1)
 C2   = -(C1^2) *(mm*(-4*eta^2 + 20*eta + 8)/(1 - eta)^5 + (1 -
  mm)*(2*eta^3 +
 12*eta^2 - 48*eta + 40)/((1 - eta)*(2 - eta))^3)

#number density of molecules
rho   = 6*eta/PI/sum_i(x[i]*m[i]*d[i]^3)

zeta{0} = PI/6*rho*sum_i(x[i]*m[i]*d[i]^0)
zeta{1} = PI/6*rho*sum_i(x[i]*m[i]*d[i]^1)
zeta{2} = PI/6*rho*sum_i(x[i]*m[i]*d[i]^2)
zeta{3} = PI/6*rho*sum_i(x[i]*m[i]*d[i]^3)

zmo  = 1 - eta

#residual contribution of the hard-sphere fluid to the compressibility factor
Zhs    = zeta{3}/zmo + 3*zeta{1}*zeta{2}/(zeta{0}*zmo^2) + (3*zeta{2}^3 -
zeta{3}*zeta{2}^3)/(zeta{0}*zmo^3)

#radial distribution function of the hard-sphere fluid (gij_hs)
ghs{0}[i] =1/zmo + 3*dd{0}[i]*zeta{2}/zmo^2 + 2*(dd{0}[i]^2)*(zeta{2}^2)/(zmo^3)
ghs{1}[i] =1/zmo + 3*dd{1}[i]*zeta{2}/zmo^2 + 2*(dd{1}[i]^2)*(zeta{2}^2)/(zmo^3)

#dgii_hs
ghs{i}_0 = ghs{0}_0
ghs{i}_1 = ghs{1}_1

#rho*dgij_hs/drho
dghs{0}[i] = zeta{3}/zmo^2 + dd{0}[i]*(3*zeta{2}/zmo^2 + 6*zeta{2}*zeta{3}/zmo^3)
```

```
+ dd{0}[i]^2*(4*zeta{2}^2/zmo^3 + 6*zeta{2}^2*zeta{3}/zmo^4)
dghs{1}[i] = zeta{3}/zmo^2 + dd{1}[i]*(3*zeta{2}/zmo^2 + 6*zeta{2}*zeta{3}/zmo^3)
+ dd{1}[i]^2*(4*zeta{2}^2/zmo^3 + 6*zeta{2}^2*zeta{3}/zmo^4)

#rho*dgii_hs/drho
dghs{i}_0 = dghs{0}_0
dghs{i}_1 = dghs{1}_1

#residual hard-chain contribution to the compressibility factor
Zhc = mm*Zhs -  sum_i(x[i]*(m[i] - 1)*(1/ghs{i}[i])*dghs{i}[i])

#derivative of I1 and I2 with respect to eta respectively
dpI1eta = a{0}*eta^0 + 2*a{1}*eta^1 + 3*a{2}*eta^2 + 4*a{3}*eta^3 +
 5*a{4}*eta^4 +
6*a{5}*eta^5 + 7*a{6}*eta^6
dpI2eta = b{0}*eta^0 + 2*b{1}*eta^1 + 3*b{2}*eta^2 + 4*b{3}*eta^3 +
 5*b{4}*eta^4
+ 6*b{5}*eta^5 + 7*b{6}*eta^6

Z1 = -2*PI*rho*dpI1eta*m2es3_m
Z2 = -PI*rho*mm*(C1*dpI2eta + C2*eta*I2)*m2e2s3_m

#dispersion contribution to the compressibility factor
Zdisp = Z1 + Z2

#compressibility factor
Z = 1 + Zhc + Zdisp

#calculate pressure
Pcal = Z*kBOLTZ*T*rho*1e30
0 = Pcal - Psys

#molar density
rhom = rho/NAV*(1e10)^3*(1e-3)


#************************************************
#************************************************
#partial derivative of number density of molecules with
respect to mole fraction
pzeta{0}xk[i] = PI/6*rho*m[i]*d[i]^0
pzeta{1}xk[i] = PI/6*rho*m[i]*d[i]^1
pzeta{2}xk[i] = PI/6*rho*m[i]*d[i]^2
pzeta{3}xk[i] = PI/6*rho*m[i]*d[i]^3
```

```
#partial derivative of first-order dispersion term for chain molecules
with respect to mole fraction

pa{0}xk[i] = m[i]*a{1}{0}/mm^2 + m[i]*(3 - 4/mm)*a{2}{0}/mm^2
pa{1}xk[i] = m[i]*a{1}{1}/mm^2 + m[i]*(3 - 4/mm)*a{2}{1}/mm^2
pa{2}xk[i] = m[i]*a{1}{2}/mm^2 + m[i]*(3 - 4/mm)*a{2}{2}/mm^2
pa{3}xk[i] = m[i]*a{1}{3}/mm^2 + m[i]*(3 - 4/mm)*a{2}{3}/mm^2
pa{4}xk[i] = m[i]*a{1}{4}/mm^2 + m[i]*(3 - 4/mm)*a{2}{4}/mm^2
pa{5}xk[i] = m[i]*a{1}{5}/mm^2 + m[i]*(3 - 4/mm)*a{2}{5}/mm^2
pa{6}xk[i] = m[i]*a{1}{6}/mm^2 + m[i]*(3 - 4/mm)*a{2}{6}/mm^2

#partial derivative of second-order dispersion term for chain molecules
with respect to mole fraction

pb{0}xk[i] = m[i]*b{1}{0}/mm^2 + m[i]*(3 - 4/mm)*b{2}{0}/mm^2
pb{1}xk[i] = m[i]*b{1}{1}/mm^2 + m[i]*(3 - 4/mm)*b{2}{1}/mm^2
pb{2}xk[i] = m[i]*b{1}{2}/mm^2 + m[i]*(3 - 4/mm)*b{2}{2}/mm^2
pb{3}xk[i] = m[i]*b{1}{3}/mm^2 + m[i]*(3 - 4/mm)*b{2}{3}/mm^2
pb{4}xk[i] = m[i]*b{1}{4}/mm^2 + m[i]*(3 - 4/mm)*b{2}{4}/mm^2
pb{5}xk[i] = m[i]*b{1}{5}/mm^2 + m[i]*(3 - 4/mm)*b{2}{5}/mm^2
pb{6}xk[i] = m[i]*b{1}{6}/mm^2 + m[i]*(3 - 4/mm)*b{2}{6}/mm^2

#partial derivative of first-order perturbation integral
pI1aux0[i] = a{0}*0*pzeta{3}xk[i]*eta^(0 - 1) + pa{0}xk[i]*eta^0
pI1aux1[i] = a{1}*1*pzeta{3}xk[i]*eta^(1 - 1) + pa{1}xk[i]*eta^1
pI1aux2[i] = a{2}*2*pzeta{3}xk[i]*eta^(2 - 1) + pa{2}xk[i]*eta^2
pI1aux3[i] = a{3}*3*pzeta{3}xk[i]*eta^(3 - 1) + pa{3}xk[i]*eta^3
pI1aux4[i] = a{4}*4*pzeta{3}xk[i]*eta^(4 - 1) + pa{4}xk[i]*eta^4
pI1aux5[i] = a{5}*5*pzeta{3}xk[i]*eta^(5 - 1) + pa{5}xk[i]*eta^5
pI1aux6[i] = a{6}*6*pzeta{3}xk[i]*eta^(6 - 1) + pa{6}xk[i]*eta^6

pI1xk[i] = pI1aux0[i] + pI1aux1[i] + pI1aux2[i] + pI1aux3[i] + pI1aux4[i]
+ pI1aux5[i] + pI1aux6[i]

#partial derivative of second-order perturbation integral
pI2aux0[i] = b{0}*0*pzeta{3}xk[i]*eta^(0 - 1) + pb{0}xk[i]*eta^0
pI2aux1[i] = b{1}*1*pzeta{3}xk[i]*eta^(1 - 1) + pb{1}xk[i]*eta^1
pI2aux2[i] = b{2}*2*pzeta{3}xk[i]*eta^(2 - 1) + pb{2}xk[i]*eta^2
pI2aux3[i] = b{3}*3*pzeta{3}xk[i]*eta^(3 - 1) + pb{3}xk[i]*eta^3
pI2aux4[i] = b{4}*4*pzeta{3}xk[i]*eta^(4 - 1) + pb{4}xk[i]*eta^4
pI2aux5[i] = b{5}*5*pzeta{3}xk[i]*eta^(5 - 1) + pb{5}xk[i]*eta^5
pI2aux6[i] = b{6}*6*pzeta{3}xk[i]*eta^(6 - 1) + pb{6}xk[i]*eta^6

pI2xk[i] = pI2aux0[i] + pI2aux1[i] + pI2aux2[i] + pI2aux3[i] +
pI2aux4[i] + pI2aux5[i] + pI2aux6[i]
```

```
#partial derivative of C1 with respect to mole fraction
pC1xk[i] = C2*pzeta{3}xk[i] - C1^2*(m[i]*(8*eta - 2*eta^2)/zmo^4
- m[i]*(20*eta - 27*eta^2 + 12*eta^3 -2*eta^4)/(zmo*(2 - eta))^2)

#partial derivative of pm2es3_m with respect to mole fraction
sum{k}_0= sum_i(x[i]*m[i]*epsilonk{0}[i]/T*sigma{0}[i]^3)
sum{k}_1= sum_i(x[i]*m[i]*epsilonk{1}[i]/T*sigma{1}[i]^3)

pm2es3mxk[i] = 2*m[i]*sum{k}[i]

#partial derivative of pm2e2s3_m with respect to mole fraction
sum{k}_0= sum_i(x[i]*m[i]*(epsilonk{0}[i]/T)^2*sigma{0}[i]^3)
sum{k}_1= sum_i(x[i]*m[i]*(epsilonk{1}[i]/T)^2*sigma{1}[i]^3)

pm2e2s3mxk[i] = 2*m[i]*sum{k}[i]

#residual contribution of the hard-sphere fluid to the Helmholtz free energy
ahs = 1/zeta{0}*(3*zeta{1}*zeta{2}/zmo + zeta{2}^3/(zeta{3}*zmo^2) +
(zeta{2}^3/zeta{3}^2 - zeta{0})*ln(zmo))

#first and second order contributions to the dispersion contribution
A1kTN = -2*PI*rho*I1*m2es3_m
A2kTN = -PI*rho*mm*C1*I2*m2e2s3_m

#dispersion contribution to the Helmholtz free energy
adisp = A1kTN + A2kTN

#residual hard-chain contribution to the Helmholtz free energy
sigma{i}_0 = sigma{0}_0
sigma{i}_1 = sigma{1}_1

ahc = mm*ahs -  sum_i( x[i]* (m[i] - 1)*ln( ghs{i}[i] ))

#residual Helmholtz free energy
ares = ahc + adisp


#partial derivative of adisp with respect to mole fraction
padispxk[i] = -2*PI*rho*(pI1xk[i]*m2es3_m + I1*pm2es3mxk[i]) -
PI*rho*((m[i]*C1*I2 + mm*pC1xk[i]*I2 + mm*C1*pI2xk[i])*m2e2s3_m +
mm*C1*I2*pm2e2s3mxk[i])

#partial derivative of gij_hs with respect to mole fraction
pghsij0xk[i] = pzeta{3}xk[i]/zmo^2 + d_0*d_0/(d_0 + d_0)*(3*pzeta{2}xk[i]/zmo^2 +
```

```
6*zeta{2}*pzeta{3}xk[i]/zmo^3) + (d_0*d_0/(d_0 +
d_0))^2*(4*zeta{2}*pzeta{2}xk[i]/zmo^3 + 6*zeta{2}^2*pzeta{3}xk[i]/zmo^4)
pghsij1xk[i] = pzeta{3}xk[i]/zmo^2 + d_1*d_1/(d_1 + d_1)*(3*pzeta{2}xk[i]/zmo^2 +
6*zeta{2}*pzeta{3}xk[i]/zmo^3) + (d_1*d_1/(d_1 +
d_1))^2*(4*zeta{2}*pzeta{2}xk[i]/zmo^3 + 6*zeta{2}^2*pzeta{3}xk[i]/zmo^4)

pghsxk0ij[i] = pzeta{3}xk_0/zmo^2 + d[i]*d[i]/(d[i] + d[i])*(3*pzeta{2}xk_0/zmo^2 +
6*zeta{2}*pzeta{3}xk_0/zmo^3) + (d[i]*d[i]/(d[i] +
d[i]))^2*(4*zeta{2}*pzeta{2}xk_0/zmo^3 + 6*zeta{2}^2*pzeta{3}xk_0/zmo^4)
pghsxk1ij[i] = pzeta{3}xk_1/zmo^2 + d[i]*d[i]/(d[i] + d[i])*(3*pzeta{2}xk_1/zmo^2 +
6*zeta{2}*pzeta{3}xk_1/zmo^3) + (d[i]*d[i]/(d[i] +
d[i]))^2*(4*zeta{2}*pzeta{2}xk_1/zmo^3 + 6*zeta{2}^2*pzeta{3}xk_1/zmo^4)

#partial derivative of ahs with respect to mole fraction
pahsxkaux0[i] = -pzeta{0}xk[i]/zeta{0}*ahs
pahsxkaux1[i] = 3*(pzeta{1}xk[i]*zeta{2} + zeta{1}*pzeta{2}xk[i])/zmo
pahsxkaux2[i] = 3*zeta{1}*zeta{2}*pzeta{3}xk[i]/zmo^2
pahsxkaux3[i] = 3*zeta{2}^2*pzeta{2}xk[i]/zeta{3}/zmo^2
pahsxkaux4[i] = zeta{2}^3*pzeta{3}xk[i]*(3*zeta{3} -1)/zeta{3}^2/zmo^3
pahsxkaux5[i] = ((3*zeta{2}^2*pzeta{2}xk[i]*zeta{3} -
2*zeta{2}^3*pzeta{3}xk[i])/zeta{3}^3 - pzeta{0}xk[i])*ln(zmo)
pahsxkaux6[i] = (zeta{0} - zeta{2}^3/zeta{3}^2)*pzeta{3}xk[i]/zmo

pahsxk[i] =  pahsxkaux0[i] + 1/zeta{0}*(pahsxkaux1[i] + pahsxkaux2[i] +
pahsxkaux3[i] + pahsxkaux4[i] + pahsxkaux5[i] + pahsxkaux6[i])

#partial derivative of ahc with respect to mole fraction
sumaux_0 = sum_i(x[i]*(m[i] - 1)/ghs{i}[i]*pghsxk0ij[i])
sumaux_1 = sum_i(x[i]*(m[i] - 1)/ghs{i}[i]*pghsxk1ij[i])

pahcxk[i] = m[i]*ahs + mm*pahsxk[i] - sumaux[i] - (m[i] - 1)*ln(ghs{i}[i])

#partial derivative of ares with respect to mole fraction
paresxk[i] = pahcxk[i] + padispxk[i]

#residual chemical potential
Sparesxk = sum_i(x[i]*paresxk[i] )

mureskT[i] = ares + (Z - 1) + paresxk[i] - Sparesxk

#Fugacity coefficient
lnPHI[i] = mureskT[i] - ln(Z)
PHI[i]   = exp(lnPHI[i])
```

D.1.2 Dynamic Parameter Optimisation

```
#Dynamic Optimisation Problem
#Catalytic Cracking of Gas Oil.
#**********************
#MSC                  *
#CAPEC, DTU, DK       *
#11.04.05             *
#**********************
# This problem describes an overall reaction of catalytic
  cracking of gas oil (A)
# to gasoline (Q) and other side products (S).
#              k1
#       A -------> Q
#        \       /
#      k3 \    / k4
#          \  /
#            S
#
# Only the concentration of A an Q were measured,
  therefore, the concentration
# of S does not appear in the model for estimation.

#The model
  dz1 = - (k1 + k3)*z1^2
  dz2 =   k1*z1^2 - k2*z2

 ;Where z1 =[A] and z2= [Q]

#Objective function.
 ;Ordinary least square(implicit)
```

D.1.3 Simple units and process models

D.1.3.1 Mixer

```
#************************
#*MSC & ERJ              *
#*CAPEC, DTU, DK         *
#*13.09.05               *
#************************
; 0 -> Benzene ; 1 -> Ethylbenzene ; 2 -> Di-ethylbenzene
; 3 -> Tri-ethylbenzene ; 4 -> Ethylene

#*****************************************
```

```
#*<<<    Mixer Process Model       >>>     *
#*****************************************

# Mass Balances

ft{1}   = sum_i(f{1}[i])
x{1}[i] = f{1}[i] / ft{1}
H{1} = sum_i(x{1}[i]*Cp[i])*(T{1}-Tref)*ft{1}
P{1} = P1 / 1.0

; Ethylene stream
ft{2}   = sum_i(f{2}[i])
y{2}[i] = f{2}[i] / ft{2}
H{2} = sum_i(y{2}[i]*Cp[i])*(T{2}-Tref)*ft{2}
P{2} = P2 / 1.0

; Vapour flowrate from Flash     (7 ->4)
ft{4}   = sum_i(f{4}v[i])
y{4}[i] = f{4}v[i] / ft{4}
H{4} = sum_i(y{4}[i]*Cp[i])*(T{4}-Tref)*ft{4}
P{4} = P4 / 1.0

; Top flowrate from B Column     (9 ->5)
ft{5}   = sum_i(f{5}l[i])
x{5}[i] = f{5}l[i] / ft{5}
H{5} = sum_i(x{5}[i]*Cp[i])*(T{5}-Tref)*ft{5}
P{5} = P5 / 1.0

; Bottom flowrate from EB Column (13->6)
ft{6}  = sum_i(f{6}l[i])
x{6}[i]= f{6}l[i]/ft{6}
H{6} = sum_i(x{6}[i]*Cp[i])*(T{6}-Tref)*ft{6}
P{6} = P6 / 1.0

0 = ft{4}    + ft{5}    + ft{6}    + ft{1}   + ft{2}   - ft{3}
0 = f{4}v[i] + f{5}l[i] + f{6}l[i] + f{1}[i] + f{2}[i] - f{3}[i]

#  Energy Balance
HM1 = sum_i(x{1}[i]*Cp[i])*(T{1}-T{3})*ft{1}
HM2 = sum_i(y{2}[i]*Cp[i])*(T{2}-T{3})*ft{2}
HM7 = sum_i(y{4}[i]*Cp[i])*(T{4}-T{3})*ft{4}
HM9 = sum_i(x{5}[i]*Cp[i])*(T{5}-T{3})*ft{5}
HM13= sum_i(x{6}[i]*Cp[i])*(T{6}-T{3})*ft{6}

0 = HM1 + HM2 + HM7 + HM9 + HM13
```

```
H{3} = sum_i(f{3}[i]*Cp[i])*(T{3}-Tref)
P{3} = P{1}
```

D.1.3.2 Reactor

```
#************************
#*MSC & ERJ             *
#*CAPEC, DTU, DK        *
#*13.09.05              *
#************************

ft{1} = sum_i(f{1}[i])
P{1}  = Patm * 760.0
H{1}  = sum_i(f{1}[i]*(T{1}-Tref))


#*****************************************
#*<<<    Reactor Process Model       >>>     *
#*****************************************
# Mass Balances
R1 = -k1f*z{2}_0 + k1b*z{2}_1
R2 = -k2f*z{2}_1 + k2b*z{2}_2
R3 = -k3f*z{2}_2
f{2}[i] = z{2}[i]*ft{2}

0 = f{1}_0 - f{2}_0 + ( R1   )*Vr*Ctr    + z{2}_0*0.0 +  0*(Tcwout+beta4+T{2})
0 = f{1}_1 - f{2}_1 + (-R1+R2)*Vr*Ctr    + z{2}_1*0.0
0 = f{1}_2 - f{2}_2 + (-R2+R3)*Vr*Ctr    + z{2}_2*0.0
0 = f{1}_3 - f{2}_3 + (-R3   )*Vr*Ctr    + z{2}_3*0.0
0 = ft{1}  - ft{2}  + (R1+R2+R3)*Vr*Ctr  + ft{2} *0.0
z{2}_4 = 1.0 - (z{2}_0 + z{2}_1 + z{2}_2 + z{2}_3)

#  Equilibrium
Psat[i] = exp(A[i] + B[i]/(T{2}-273.15 + C[i]))
Psat_3  = 10^(A_3  + B_3 /(T{2}-273.15 + C_3))
K[i] = Psat[i] / P{1}

fun[i] = z{2}[i]*(K[i]-1.0)/(beta4*K[i]+1.0-beta4)
0 = sum_i(fun[i]) + beta4*0.0

# Energy Balance
DHr1 = (-Cp_0+Cp_1)*(T{2}-Tref)
DHr2 = (-Cp_1+Cp_2)*(T{2}-Tref)
```

```
DHr3 = (-Cp_3)     *(T{2}-Tref)
HR   = DHr1*R1 + DHr2*R2 + DHr3*R3
Cpav = sum_i(z{2}[i]*Cp[i])
deltaT1 = T{2} - Tcwin
deltaT2 = T{2} - Tcwout
Qr = UAr  * (deltaT1-deltaT2)/(ln(deltaT1/deltaT2))

0 = ft{2}*Cpav*(T{1}-T{2}) - HR - Qr + T{2}*0.0
0 = UAr*((deltaT1-deltaT2)/ln(deltaT1/deltaT2))
 - mcwr*Cpcw*(Tcwout-Tcwin) + Tcwout*0.0

ft{2}v = beta4*ft{2}
ft{2}l = (1.0-beta4)*ft{2}
x{2}[i] = z{2}[i] / (1.0-beta4*(1.0-K[i]))
y{2}[i] = K[i]*x{2}[i]
P{2}    = P{1} / 760.0
H{2}    = sum_i(f{2}[i]*(T{2}-Tref))
```

D.1.3.3 Flash

```
#************************
#*MSC & ERJ             *
#*CAPEC, DTU, DK        *
#*13.09.05              *
#************************

#*****************************************
#*<<<     Flash Process Model      >>>     *
#*****************************************

; Vapour pressures (Antoine)
Psat[i] = exp(A[i] + B[i]/(T{1} + C[i]))
Psat_3  = 10^(A_3 +  B_3 /(T{1} + C_3 ))

; Equilibrium
K[i] = Psat[i] / P{1}
K_3  = Psat_3  / P{1}

#  Mass Balances (Isothermal Flash)
; Feed Stream
ft{1}  = sum_i(f{1}[i])
x{1}[i] = f{1}[i] / ft{1}
H{1}  = 5
```

```
fun[i] = x{1}[i]*(K[i]-1.0) / (beta*K[i]+1.0-beta)
0      = sum_i(fun[i])

; Output streams
ft{3} = (1.0-beta)*ft{1}
ft{2} = beta*ft{1}

x{3}[i]= x{1}[i]/(1.0-beta*(1.0-K[i]))
T{3} = T{1} + 273.15
P{3} = P{1} / 760.0
H{3} = 2.5

y{2}[i]= K[i]*x{3}[i]
T{2} = T{1} + 273.15
P{2} = P{1} / 760.0
; H{2} = 2.5

0 = f{3}[i] - x{1}[i]/(1.0-beta*(1.0-K[i])) * ft{3}
0 = f{2}[i] - K[i]*x{3}[i] * ft{2}
0 = H{1} - (H{2} + H{3})
```

D.1.3.4 Benzene column

```
#************************
#*MSC & ERJ              *
#*CAPEC, DTU, DK         *
#*13.09.05               *
#************************

#*****************************************
#*<<<     Benzene Column      >>>         *
#*****************************************


#  Mass Balances
P{1}   = Pt * 760.0
T{1}   = TK
H{1}   = 5.0
ft{1}  = sum_i(f{1}[i])

f{2}_0  = LKRD_BC*f{1}_0
f{2}_1  = HKRD_BC*f{1}_1
f{2}_2  = 0.0
f{2}_3  = 0.0
f{2}_4  = f{1}_4
```

```
ft{2}   = sum_i(f{2}[i])
x{2}[i] = f{2}[i] / ft{2}

ft{3}  = ft{1} - ft{2}
f{3}_0 = (1.0-LKRD_BC)*f{1}_0
f{3}_1 = (1.0-HKRD_BC)*f{1}_1
f{3}_2 = f{1}_2
f{3}_3 = f{1}_3
f{3}_4 = 0.0
x{3}[i] = f{3}[i] / ft{3}

#   Condenser
V1BC = ft{2}*(ReBC+1.0)

#   Reboiler

#  Energy Balance
   ;Distillate's Dew point
Psat9[i] = exp(A[i] + B[i]/(Tdist + C[i]))
Psat9_3  = 10^(A_3  + B_3 /(Tdist + C_3 ))
dew[i]   = x{2}[i]*P{1} / Psat9[i]
0 = 1.0 - sum_i(dew[i])

T{2} = Tdist + 273.15
P{2} = P{1}  / 760.00
H{2} = 5.0

;    Bottom's Bubblepoint
P{3} = P{1}
Psat10[i] = exp(A[i] + B[i]/(Tbub + C[i]))
Psat10_3  = 10^(A_3  + B_3 /(Tbub + C_3 ))
bub[i] = x{3}[i]*Psat10[i] / P{1}
0 = 1.0 - sum_i(bub[i])

T{3} = Tbub + 273.15
P{3} = P{1} / 760.00
H{3} = 5.0
```

D.1.3.5 Ethyl Benzene column

```
#************************
#*MSC & ERJ             *
#*CAPEC, DTU, DK        *
#*13.09.05              *
#************************
```

```
;************************************
;*<<<<     Ethylbenzene Column    >>>*
;************************************
; 10->1, 11->2, 12->3

#  Mass Balances
    ;Inlet stream
ft{1} = sum_i(f{1}[i])
T{1}  = TK
P{1}  = Patm * 760.0
H{1}  = 5.0

; Distillate's flowrate
f{2}_0 = f{1}_0
f{2}_1 = CLRD_EBC*f{1}_1
f{2}_2 = HKRD_EBC*f{1}_2
f{2}_3 = 0.0
f{2}_4 = 0.0
ft{2}  = sum_i(f{2}[i])
x{2}[i] = f{2}[i] / ft{2}

; Bottom's flowrate
ft{3} = ft{1} - ft{2}
f{3}_0 = 0.0
f{3}_1 = (1.0-CLRD_EBC)*f{1}_1
f{3}_2 = (1.0-HKRD_EBC)*f{1}_2
f{3}_3 = f{1}_3
f{3}_4 = 0.0
x{3}[i] = f{3}[i] / ft{3}

;   Condenser
V1_EBC = ft{2}*(Re_EBC+1.0D0)
;   Reboiler

# Energy Balance
;   Distillate's Dewpoint
Psatd[i] = exp(A[i] + B[i]/(Tdew-273.15+C[i]))
dew[i] = x{2}[i]*P{1} / Psatd[i]
0 = 1.0 - sum_i(dew[i])

T{2} = Tdew
P{2} = P{1} / 760.0
H{2} = 5.0
```

```
;   Bottom's Bubblepoint
Psatb[i] = exp(A[i] + B[i]/(Tbub-273.15+C[i]))
bub[i] = x{3}[i]*Psatb[i] / P{1}
0 = 1.0 - sum_i(bub[i])

T{3} = Tbub
P{3} = P{1} / 760.0
H{3} = 5.0
```

D.2 Case Study

D.2.1 Short-Path evaporator

```
#Short Path Evaporator
#************************
#*MSC                    *
#*CAPEC, DTU, DK         *
#*10.03.04               *
#************************

#Var definition
 ;P     -System Pressure [Pa]
 ;T     -Temperature profile in the liquid film [K]
 ;Tf    -Feed Temperature [K]
 ;Ts    -Surface temperature [K]
 ;Tw    -Wall Temperature [=] {K} (Heating jacket )
 ;Flow  -Feed Flow [Kg/h]

    ;Pressure of saturated vapor
     Pvap[i] = exp(DB_AntoineA[i] - DB_AntoineB[i]/(Ts + DB_AntoineC[i])

    ;Heat of vaporization [J/kmol]
      DHvap[i] = ADippr103[i]*(1.0 - (Ts/Tc[i]))^(BDippr103[i] +
       CDippr103[i]*(Ts/Tc[i]) + DDippr103[i]*(Ts/Tc[i])^2)

    ;Liquid Heat Capacity [J/(kmol*K)]
     Cp[i] = ADippr105[i] + BDippr105[i]*Ts + CDippr105[i]*Ts^2 +
      DDippr105[i]*Ts^3 + EDippr105[i]*Ts^4

    ;Density [kmol/m3]
     rho[i] = ADippr101[i]/BDippr101[i]^(1.0 + ( 1.0 -
      Ts/CDippr101[i])^DDippr101[i])

    ;Liquid Viscosity [Kg/m s]
```

```
     Eta[i] = exp(ADippr108[i]+ BDippr108[i]/Ts + CDippr108[i]*ln(Ts) +
      DDippr108[i]*Ts^EDippr108[i])

    ;Thermal Conductivity [W/(m K)]
     Lambda[i] = ADippr111[i] + BDippr111[i]*Ts + CDippr111[i]*Ts^2

    ;Kinematic Viscosity  [m2/s]
      nu[i]  = Eta[i]/rho[i]

    ;Total concentration mol/m3
     Q[i] = Flow[i]*MW[i]/rho[i]
     C[i] = Flow[i]/Qtot
     Qtot = sum_i(Q[i])
     Ctot = sum_i(C[i])

     ;Feed, molar flow [mol/s]
       FlowTot = sum_i(Flow[i])

    ;Mixture properties
    x[i]      = Flow[i]/FlowTot
    rhoMix    = sum_i(x[i]*rho[i])
    LambdaMix = sum_i(x[i]*Lambda[i])
    CpMix     = sum_i(x[i]*Cp[i])
    MWMix     = sum_i(x[i]*MW[i])
    EtaMix    = sum_i(x[i]*Eta[i])
    DHvapMix  = sum_i(x[i]*DHvap[i])
    nuMix     = EtaMix/rhoMix

    Gamma  = 70.0
    Flux0  = 2.0E0*PI*r1*Gamma*rhoMix/(3.6E05*MWMix)

#*****************************************
  ;Evaporation rate efficiency
  ;number of intermolecualr collisions before the vapor reaches isotropic
   state
   nmc  = 5
  ;mean path of vapor molecula [m]
   beta = d/4.0
  ;Surafce ratio
   F    = Ak/(Ak + Ah)
  ;Degree of anisotropy of the vapor phase in the space between the
   evaporator and condenser
   kappa = 10.0^(0.2*F + 1.38*(F + 0.1)^4.0)
   EREF  = 1.0 - (1.0 - F)*(1.0 - exp (- d / kappa/beta))^nmc
  ;Pression system effect
```

```
   EREF  = EREF*(P/Pref)
#*********************************************
 ;Integration parameter
 ;Number of radial integration points.
  N   = 10
  ay  = 0.0

#*********************************************
    h1     = (3.0*nuMix*FlowTot/(2.0*PI*r1*g*Ctot))^(1.0/3.0)
    by     = h1
    Dy     = (by - ay)/N

    Theta = MWMix*LambdaMix/(rhoMix*CpMix)
    k[i]  = Act[i]*Pvap[i]/sqrt(2.0*PI*Rg*MW[i]*Ts)
    ktot  = sum_i(x[i]*k[i])

    Y[i]   = i*Dy
    Vz[i]  = g*h1^2/nuMix*(Y[i]/h1 - 0.5*Y[i]*Y[i]/h1^2)


   ;BC at y = 1
     0   = T_10 - T_9 + Dy*DHvapMix*ktot/LambdaMix

   #Film surface temperature
    Ts     = T_10

  #Generate the ODEs for integrating
  ;BC at y = 0
     dT_0 = 0.0

    dT[j]  =(Theta/Vz[j])*(T[j+1]  - 2.0*T[j]  + T[j-1]/Dy/Dy

#Rate of evaporation (kmol/m2.h)
 dFlow[i] = - 2.0*PI*r1*k[i]*x[i]* EREF
```

D.2.2 Polimerisation of MMA

D.2.2.1 Steady-State model

```
#MMA POLYMERISATION REACTOR
#STEADY STATE OF THE NONLINEAR MODEL
#***********************
#*MSC & TLA             *
#*CAPEC, DTU, DK        *
#*10.07.05              *
```

```
#***********************

; Kinetic expressions
   kp  = Ap*exp (-Ep/R/T)
   kfm = Afm*exp(-Efm/R/T)
   kI  = AI*exp (-EI/R/T)
   ktd = Atd*exp(-Etd/R/T)
   ktc = Atc*exp(-Etc/R/T)

   P0 = sqrt(2.0*fe*CI*kI/(ktd+ktc))

   rP = kp*Cm*P0
   rT = kfm*Cm*P0
   rI = kI*CI

; Nonlinear model in steady state
   0 = F*(Cmin-Cm)/V - (rP + rT)
   0 = (FI*CIin-F*CI)/V - rI
   0 = mDH*rP/(rho*Cp) - U*A*(T-Tj)/(rho*Cp*V) + F*(Tin-T)/V
   0 = Fcw*(Tw0-Tj)/V0 + U*A*(T-Tj)/(rhow*Cpw*V0)
```

D.2.2.2 Dynamic version of the MMA model

```
#MMA POLYMERISATION REACTOR
#DYNAMIC SIMULATION
#***********************
#*MSC & TLA              *
#*CAPEC, DTU, DK         *
#*10.07.05               *
#***********************

; Kinetic expressions
   kp  = Ap*exp (-Ep/R/T)
   kfm = Afm*exp(-Efm/R/T)
   kI  = AI*exp (-EI/R/T)
   ktd = Atd*exp(-Etd/R/T)
   ktc = Atc*exp(-Etc/R/T)

   P0 = sqrt(2.0*fe*CI*kI/(ktd+ktc))

   rP = kp*Cm*P0
   rT = kfm*Cm*P0
   rI = kI*CI
```

```
; Nonlinear model in steady state
   dCm = F*(Cmin-Cm)/V - (rP + rT)
   dCI = (FI*CIin-F*CI)/V - rI
   dT  = mDH*rP/(rho*Cp) - U*A*(T-Tj)/(rho*Cp*V) + F*(Tin-T)/V
   dTj = Fcw*(Tw0-Tj)/V0 + U*A*(T-Tj)/(rhow*Cpw*V0)
```

D.2.2.3 Bifurcation analysis of the MMA model

```
#MMA POLYMERISATION REACTOR
#BIFURCATION ANALYSIS
#************************
#*MSC & TLA             *
#*CAPEC, DTU, DK        *
#*10.07.05              *
#************************

; Continuation parameters
   FI  = cont_alpha
   Tw0 = cont_beta

; Kinetic expressions
   kp  = Ap*exp (-Ep/R/T)
   kfm = Afm*exp(-Efm/R/T)
   kI  = AI*exp (-EI/R/T)
   ktd = Atd*exp(-Etd/R/T)
   ktc = Atc*exp(-Etc/R/T)

   P0 = sqrt(2.0*fe*CI*kI/(ktd+ktc))

   rP = kp*Cm*P0
   rT = kfm*Cm*P0
   rI = kI*CI

; Nonlinear model in steady state
   0 = F*(Cmin-Cm)/V - (rP + rT)
   0 = (FI*CIin-F*CI)/V - rI
   0 = mDH*rP/(rho*Cp) - U*A*(T-Tj)/(rho*Cp*V) + F*(Tin-T)/V
   0 = Fcw*(Tw0-Tj)/V0 + U*A*(T-Tj)/(rhow*Cpw*V0)
```

D.2.2.4 Closed loop of the linearized model of MMA

```
#MMA POLYMERISATION REACTOR
#CLOSED LOOP SIMULATION
```

```
#************************
#*MSC & TLA              *
#*CAPEC, DTU, DK         *
#*10.07.05               *
#************************

# Steady State 2 (unstable)
Cmss = 5.88881
CIss = 0.02473
Tss  = 353.419
Tjss = 334.357

# Control input at the steady state
FI_ss   = 0.0032
FCW_ss  = 0.1588

# Unperturbed Disturbances
Cmin_ss = 6.4678
CIin_ss = 8.0
Tin_ss  = 350.0
TW0_ss  = 293.2

# Perturbed disturbances
Cmin_per = 6.4678
CIin_per = 8.0
Tin_per  = 350.0
TW0_per  = 293.2

#************************
#Jacobian
; inputs
Cmin = Cmin_ss
CIin = CIin_ss
Tin  = Tin_ss
TW0  = TW0_ss
FI   = FI_ss
FCW  = FCW_ss

;States
Cm = Cmss
CI = CIss
T  = Tss
Tj = Tjss

;Reactions rates
```

```
kP  = Ap *exp(-EP /R/T)
kfm = Afm*exp(-Efm/R/T)
kI  = AI *exp(-EI /R/T)
ktd = Atd*exp(-Etd/R/T)
ktc = Atc*exp(-Etc/R/T)
p0  = sqrt(2.0*fe*kI*CI/(ktd + ktc))

rP = kP*Cm*p0
rT = kfm*Cm*p0
rI = kI*CI

;Jacobian evaluation
dp0_CI  = p0/2.0/CI
dp0_T   = sqrt(fe*CI/2.0)*((Atd*exp((EI - Etd)/R/T) + Atc*exp((EI -
Etc)/R/T))/AI)^(-3.0/2.0)*((Atd*exp((EI - Etd)/R/T)*(EI - Etd) +
Atc*exp((EI - Etc)/R/T)*(EI - Etc))/AI/R/T^2)


dkP_T   = kP *EP /R/T^2
dkfm_T  = kfm*Efm/R/T^2
dkI_T   = kI *EI /R/T^2
dktd_T  = ktd*Etd/R/T^2
dktc_T  = ktc*Etc/R/T^2

drP_Cm  = kP*p0
drP_CI  = kP*Cm*dp0_CI
drP_T   = Cm*(kP*dp0_T + p0*dkP_T)
drT_Cm  = kfm*p0
drT_CI  = kfm*Cm*dp0_CI
drT_T   = Cm*(kfm*dp0_T + p0*dkfm_T)

dfm_Cm  = - F/V - (drP_Cm + drT_Cm)
dfm_CI  = - (drP_CI + drT_CI)
dfm_T   = - (drP_T  + drT_T )
dfm_Tj  = 0.0

dfI_Cm  = 0.0
dfI_CI  = - (F/V + kI)
dfI_T   = - CI*dkI_T
dfI_Tj  = 0.0

dfT_Cm  = mDH*drP_Cm/(rho*Cp)
dfT_CI  = mDH*drP_CI/(rho*Cp)
dfT_T   = - F/V - U*A/(rho*Cp*V) + mDH*drP_T/(rho*Cp)
dfT_Tj  = U*A/(rho*Cp*V)
```

```
dfTj_Cm = 0.0
dfTj_CI = 0.0
dfTj_T  = U*A/(rhoW*CpW*V0)
dfTj_Tj = - FCW/V0 - U*A/(rhoW*CpW*V0)

;Matrix A
a11 = dfm_Cm
a12 = dfm_CI
a13 = dfm_T
a14 = dfm_Tj
a21 = dfI_Cm
a22 = dfI_CI
a23 = dfI_T
a24 = dfI_Tj
a31 = dfT_Cm
a32 = dfT_CI
a33 = dfT_T
a34 = dfT_Tj
a41 = dfTj_Cm
a42 = dfTj_CI
a43 = dfTj_T
a44 = dfTj_Tj

;Matrix B
b11 = 0.0
b12 = 0.0
b21 = CIin/V
b22 = 0.0
b31 = 0.0
b32 = 0.0
b41 = 0.0
b42 = (TW0 - Tj)/V0

;Matrix Bd
bd11 = F/V
bd12 = 0.0
bd13 = 0.0
bd14 = 0.0
bd21 = 0.0
bd22 = FI/V
bd23 = 0.0
bd24 = 0.0
bd31 = 0.0
bd32 = 0.0
```

```
bd33 = F/V
bd34 = 0.0
bd41 = 0.0
bd42 = 0.0
bd43 = 0.0
bd44 = FCW/V0

#Control PID

FI_p  = kc1*Cm_p + Zero*kc1/taoI1*I1
FCW_p = kc2*T_p  + Zero*kc2/taoI2*I2

Cmin_p = Cmin_per - Cmin_ss
CIin_p = CIin_per - CIin_ss
Tin_p  = Tin_per  - Tin_ss
TW0_p  = TW0_per  - TW0_ss

#Linearized model
A1 = a11*Cm_p + a12*CI_p + a13*T_p + a14*Tj_p
A2 = a21*Cm_p + a22*CI_p + a23*T_p + a24*Tj_p
A3 = a31*Cm_p + a32*CI_p + a33*T_p + a34*Tj_p
A4 = a41*Cm_p + a42*CI_p + a43*T_p + a44*Tj_p

B1 = b11*FI_p + b12*FCW_p
B2 = b21*FI_p + b22*FCW_p
B3 = b31*FI_p + b32*FCW_p
B4 = b41*FI_p + b42*FCW_p

Bd1 = bd11*Cmin_p + bd12*CIin_p + bd13*Tin_p + bd14*TW0_p
Bd2 = bd21*Cmin_p + bd22*CIin_p + bd23*Tin_p + bd24*TW0_p
Bd3 = bd31*Cmin_p + bd32*CIin_p + bd33*Tin_p + bd34*TW0_p
Bd4 = bd41*Cmin_p + bd42*CIin_p + bd43*Tin_p + bd44*TW0_p

#**************************

dCm_p = A1 + B1 + Bd1
dCI_p = A2 + B2 + Bd2
dT_p  = A3 + B3 + Bd3
dTj_p = A4 + B4 + Bd4

dI1 = Cm_p
dI2 = T_p

Cm = Cm_p + Cmss
CI = CI_p + CIss
```

```
T  = T_p  + Tss
Tj = Tj_p + Tss

FI  = FI_p + FI_ss
FCW = FCW_p + FCW_ss
```

D.2.3 Tennessee Eastman Problem

```
# Tennessee Eastman Process
#************************
#*MSC                   *
#*CAPEC, DTU, DK        *
#*10.07.05              *
#************************
# Reference:
;  [1] Tobias Jockenhvel, Lorenz T. Bigler, Andreas Wchter
;      Dynamic Optimisation of the TennesseeEastman Process
       Using OptControlCentre.
;      Computers and Chemical Engineering 27(2003), pp. 1513-1531.
;  [2] N.L Ricker and J. H. Lee
;      Nonlinear Modeling and State Estimation for the
       Tennessee Eastman Challenge Process
;      Computer Chem. Engng (1995), Vol. 19, No. 9, pp. 983-1005.
;  [3] J.J. Downs and E. F. Vogel
;      A Plant-Wide Industrial Process Control Problem
;      Computer Chem Engng. (1993), Vol. 17, No. 3, pp. 245-255

#Reaction Stoichiometric is:
;   A(g) + C(g) + D(g) -->  G(L),   Product 1
;   A(g) + C(g) + E(g) -->  H(L),   Product 2
;          A(g) + E(g) -->  F(L),   Byproduct
;                3D(g) --> 2F(L),   Byroduct 1
################################################

#*********************************************************
# Stream property data - Taken from Table 2, Downs & Vogel*
#*********************************************************

  ;Molecular weight
  M_A =  2.0
  M_B = 25.4
  M_C = 28.0
  M_D = 32.0
  M_E = 46.0
```

```
  M_F = 48.0
  M_G = 62.0
  M_H = 76.0

  ;Vapor heat capacity [KJ/(Kg C)]
  cp_vap_A = 14.6 * M_A
  cp_vap_B = 2.04 * M_B
  cp_vap_C = 1.05 * M_C
  cp_vap_D = 1.85 * M_D
  cp_vap_E = 1.87 * M_E
  cp_vap_F = 2.02 * M_F
  cp_vap_G = 0.712* M_G
  cp_vap_H = 0.628* M_H

  ;Liquid heat capacity (A-C not condensable) [KJ/(Kg C)]
  cp_liq_A = 0     * M_A
  cp_liq_B = 0     * M_B
  cp_liq_C = 0     * M_C
  cp_liq_D = 7.66  * M_D
  cp_liq_E = 4.17  * M_E
  cp_liq_F = 4.45  * M_F
  cp_liq_G = 2.55  * M_G
  cp_liq_H = 2.45  * M_H

  ;Heat of vaporization [KJ/Kg]
  H_vap_A = 0      * M_A
  H_vap_B = 0      * M_B
  H_vap_C = 0      * M_C
  H_vap_D = 202    * M_D
  H_vap_E = 372    * M_E
  H_vap_F = 372    * M_F
  H_vap_G = 523    * M_G
  H_vap_H = 486    * M_H

#************
# Constants *
#************

 ;Reference Temperature
  Tref = 273.15

 ;Universal gas constant
  ;RkJ [=][kPa m3/(kmol K)] or [kJ/(kmol K)] and Rkcal [=] kcal/(kmol K)
  RkJ   = 8.31451
  Rkcal = 1.987
```

```
#***************************
#Cooling water heat capacity*
#***************************
 cp_cw = 4.18

#Reaction stochiometric coeficients
 mue_A_1 = -1
 mue_A_2 = -1
 mue_A_3 = -1/3

 mue_B_1 =  0
 mue_B_2 =  0
 mue_B_3 =  0

 mue_C_1 = -1
 mue_C_2 = -1
 mue_C_3 =  0

 mue_D_1 = -1
 mue_D_2 =  0
 mue_D_3 = -1

 mue_E_1 =  0
 mue_E_2 = -1
 mue_E_3 = -1/3

 mue_F_1 =  0
 mue_F_2 =  0
 mue_F_3 =  1

 mue_G_1 =  1
 mue_G_2 =  0
 mue_G_3 =  0

 mue_H_1 =  0
 mue_H_2 =  1
 mue_H_3 =  0

# Antoine Constants
#P [=] Pa and T [=] C
 A_D = 20.81
 B_D = -1444
 C_D = 259.0
```

```
 A_E = 21.24
 B_E = -2114
 C_E = 265.5

 A_F = 21.24
 B_F = -2144
 C_F = 265.5

 A_G = 21.32
 B_G = -2748
 C_G = 232.9

 A_H = 22.10
 B_H = -3318
 C_H = 249.6

#****************************************
# Feed streams; Table 1 - Downs and Vogel*
#****************************************

 ;Molar feed flow [Kmol/s]
  stream_1_flow  = 11.2  / 3600
  stream_2_flow  = 114.5 / 3600
  stream_3_flow  = 98.0  / 3600
  stream_4_flow  = 417.5 / 3600
  stream_8_flow  = 1201.5/ 3600
  stream_9_flow  = 15.1  / 3600
  stream_10_flow = 259.5 / 3600
  stream_11_flow = 211.3 / 3600

  ;Temperatures ; 318.15 K

  stream_1_T = 45 + Tref
  stream_2_T = 45 + Tref
  stream_3_T = 45 + Tref
  stream_4_T = 45 + Tref

  ;Composition of Feed Stream A
  stream_1_conc_A = 0.99990
  stream_1_conc_B = 0.00010
  stream_1_conc_C = 0.00000
  stream_1_conc_D = 0.00000
  stream_1_conc_E = 0.00000
  stream_1_conc_F = 0.00000
  stream_1_conc_G = 0.00000
```

```
  stream_1_conc_H = 0.00000

 ;Composition of Feed Stream D
  stream_2_conc_A = 0.00000
  stream_2_conc_B = 0.00010
  stream_2_conc_C = 0.00000
  stream_2_conc_D = 0.99990
  stream_2_conc_E = 0.00000
  stream_2_conc_F = 0.00000
  stream_2_conc_G = 0.00000
  stream_2_conc_H = 0.00000

  ;Composition of Feed Stream E
  stream_3_conc_A = 0.00000
  stream_3_conc_B = 0.00000
  stream_3_conc_C = 0.00000
  stream_3_conc_D = 0.00000
  stream_3_conc_E = 0.99990
  stream_3_conc_F = 0.00010
  stream_3_conc_G = 0.00000
  stream_3_conc_H = 0.00000

  ;Composition of Feed Stream C
  stream_4_conc_A = 0.48500
  stream_4_conc_B = 0.00500
  stream_4_conc_C = 0.51000
  stream_4_conc_D = 0.00000
  stream_4_conc_E = 0.00000
  stream_4_conc_F = 0.00000
  stream_4_conc_G = 0.00000
  stream_4_conc_H = 0.00000

#***************************
#Feed stream heat capacities*
#***************************

stream_1_cp = stream_1_conc_A*cp_vap_A + stream_1_conc_B*cp_vap_B +
stream_1_conc_C*cp_vap_C + stream_1_conc_D*cp_vap_D + stream_1_conc_E*cp_vap_E +
stream_1_conc_F*cp_vap_F + stream_1_conc_G*cp_vap_G + stream_1_conc_H*cp_vap_H
stream_2_cp = stream_2_conc_A*cp_vap_A + stream_2_conc_B*cp_vap_B +
stream_2_conc_C*cp_vap_C + stream_2_conc_D*cp_vap_D + stream_2_conc_E*cp_vap_E +
stream_2_conc_F*cp_vap_F + stream_2_conc_G*cp_vap_G + stream_2_conc_H*cp_vap_H
stream_3_cp = stream_3_conc_A*cp_vap_A + stream_3_conc_B*cp_vap_B +
stream_3_conc_C*cp_vap_C + stream_3_conc_D*cp_vap_D + stream_3_conc_E*cp_vap_E +
stream_3_conc_F*cp_vap_F + stream_3_conc_G*cp_vap_G + stream_3_conc_H*cp_vap_H
```

```
stream_4_cp = stream_4_conc_A*cp_vap_A + stream_4_conc_B*cp_vap_B +
stream_4_conc_C*cp_vap_C + stream_4_conc_D*cp_vap_D + stream_4_conc_E*cp_vap_E +
stream_4_conc_F*cp_vap_F + stream_4_conc_G*cp_vap_G + stream_4_conc_H*cp_vap_H

#**********
#Utilities*
#**********
 ;cooling water flow mol/s
  reactor_m_CWS   = 93.37/3600*1000/18
  separator_m_CWS = 49.37/3600*1000/18

;Stripper stream flow [Kg/s] equivalent to 47.446% of stripper valve
  stripper_m_CWS  = 230.31/3600

#*****************
# Volumes  [=] m3*
#*****************
  ;Mixing zone volume
  Vm = 141.53

  ;Total Reactor volume
  Vr = 36.8117791

  ;Total separator volume
  Vs = 99.1

#**********
#Densities*
#**********
  ;Molar liquid density [=] kmol/m3
  rho_liq_reactor   = 9.337145754
  rho_liq_separator = 10.29397546

#*************************
#Activity coefficients VLE*
#*************************
 ;Reactor
  gamma_D_r = 0.996011
  gamma_E_r = 1
  gamma_F_r = 1.078
  gamma_G_r = 0.999
  gamma_H_r = 0.999

  ;Separator
  gamma_D_s = 1.001383
```

```
  gamma_E_s = 1.001383
  gamma_F_s = 1.091383
  gamma_G_s = 1.001383
  gamma_H_s = 0.992188



  alpha_1 = 1.0399157
  alpha_2 = 1.011373129
  alpha_3 = 1

#Heat transfer exchange
  UA     = 127.6
  UA_sep = 152.7

#Compressor

s_kap = .7166374645

#Stripper

 ;Density
  rho_liq_stripper = 8.6496e0

#Split factors Stripper
  phi_A = 1
  phi_B = 1
  phi_C = 1

PC_1  = 3.51e-7
PC_6  = -0.00011
PC_11 = 0.011351
PC_16 = 0.548012

PC_2  = 3.69e-7
PC_7  = -0.0001
PC_12 = 0.010197
PC_17 = 0.620794

PC_3  = 3.69e-7
PC_8  = -0.0001
PC_13 = 0.010049
PC_18 = 0.628854

PC_4  = 1.84e-9
```

```
PC_9  = 1.37e-5
PC_14 = 0.000217
PC_19 = 0.001393

PC_5  = 8.32e-8
PC_10 = -7.64e-6
PC_15 = 0.000976
PC_20 = -0.01568

###################
#From Mixer zone #
###################
# Pressure in the mixing zone

mixing_zone_N = mixing_zone_NA + mixing_zone_NB + mixing_zone_NC +
mixing_zone_ND + mixing_zone_NE + mixing_zone_NF + mixing_zone_NG +
mixing_zone_NH

 pm_MPa = mixing_zone_N * (RkJ * Tm / Vm) / 1000


  stream_6_conc_A = mixing_zone_NA / mixing_zone_N
  stream_6_conc_B = mixing_zone_NB / mixing_zone_N
  stream_6_conc_C = mixing_zone_NC / mixing_zone_N
  stream_6_conc_D = mixing_zone_ND / mixing_zone_N
  stream_6_conc_E = mixing_zone_NE / mixing_zone_N
  stream_6_conc_F = mixing_zone_NF / mixing_zone_N
  stream_6_conc_G = mixing_zone_NG / mixing_zone_N
  stream_6_conc_H = mixing_zone_NH / mixing_zone_N


######################
#From reactor zone   #
######################
# (A, B, C not condensable)
  reactor_x_A = 0.0
  reactor_x_B = 0.0
  reactor_x_C = 0.0

reactor_x_D = reactor_ND / (reactor_ND + reactor_NE + reactor_NF + reactor_NG +
reactor_NH)
  reactor_x_E = reactor_NE / (reactor_ND + reactor_NE + reactor_NF + reactor_NG +
reactor_NH)
  reactor_x_F = reactor_NF / (reactor_ND + reactor_NE + reactor_NF + reactor_NG +
reactor_NH)
```

```
  reactor_x_G = reactor_NG / (reactor_ND + reactor_NE + reactor_NF + reactor_NG +
reactor_NH)
  reactor_x_H = reactor_NH / (reactor_ND + reactor_NE + reactor_NF + reactor_NG +
reactor_NH)

Vr_liq = (reactor_ND + reactor_NE + reactor_NF + reactor_NG + reactor_NH) /
rho_liq_reactor

   Vr_vap = Vr - Vr_liq

  p_A_r = reactor_NA * RkJ * Tr /  Vr_vap / 1000
  p_B_r = reactor_NB * RkJ * Tr /  Vr_vap / 1000
  p_C_r = reactor_NC * RkJ * Tr /  Vr_vap / 1000

    pr_sat_D = 1e-3 * exp(A_D + (B_D / (C_D + Tr - Tref) ) ) / 1000
    pr_sat_E = 1e-3 * exp(A_E + (B_E / (C_E + Tr - Tref) ) ) / 1000
    pr_sat_F = 1e-3 * exp(A_F + (B_F / (C_F + Tr - Tref) ) ) / 1000
    pr_sat_G = 1e-3 * exp(A_G + (B_G / (C_G + Tr - Tref) ) ) / 1000
    pr_sat_H = 1e-3 * exp(A_H + (B_H / (C_H + Tr - Tref) ) ) / 1000

  p_D_r = gamma_D_r * reactor_x_D * pr_sat_D
  p_E_r = gamma_E_r * reactor_x_E * pr_sat_E
  p_F_r = gamma_F_r * reactor_x_F * pr_sat_F
  p_G_r = gamma_G_r * reactor_x_G * pr_sat_G
  p_H_r = gamma_H_r * reactor_x_H * pr_sat_H

   pr_MPa = p_A_r + p_B_r + p_C_r + p_D_r + p_E_r + p_F_r + p_G_r + p_H_r

#Flow rates
  press_m_r_diff = sqrt(pm_MPa  - pr_MPa)

  stream_6_flow = 0.8333711713 * press_m_r_diff

# assumption base case
  press_r_s_diff = sqrt( pr_MPa - ps_MPa)
  stream_7_flow = 1.53546206685993 * press_r_s_diff


  stream_7_conc_A  = p_A_r / pr_MPa
  stream_7_conc_B  = p_B_r / pr_MPa
  stream_7_conc_C  = p_C_r / pr_MPa
  stream_7_conc_D  = p_D_r / pr_MPa
  stream_7_conc_E  = p_E_r / pr_MPa
  stream_7_conc_F  = p_F_r / pr_MPa
  stream_7_conc_G  = p_G_r / pr_MPa
```

```
  stream_7_conc_H  = p_H_r / pr_MPa


######################
#From Separator zone #
######################
  ;(A, B, C not condensable)
  separator_x_A = 0.0
  separator_x_B = 0.0
  separator_x_C = 0.0

  separator_x_D = separator_ND / (separator_ND + separator_NE + separator_NF +
separator_NG + separator_NH)
  separator_x_E = separator_NE / (separator_ND + separator_NE + separator_NF +
separator_NG + separator_NH)
  separator_x_F = separator_NF / (separator_ND + separator_NE + separator_NF +
separator_NG + separator_NH)
  separator_x_G = separator_NG / (separator_ND + separator_NE + separator_NF +
separator_NG + separator_NH)
  separator_x_H = separator_NH / (separator_ND + separator_NE + separator_NF +
separator_NG + separator_NH)

Vs_liq = (separator_ND + separator_NE + separator_NF + separator_NG + separator_NH) /
rho_liq_separator

  Vs_vap = Vs - Vs_liq

p_A_s = separator_NA * RkJ * Ts / Vs_vap  /1000
p_B_s = separator_NB * RkJ * Ts / Vs_vap  /1000
p_C_s = separator_NC * RkJ * Ts / Vs_vap  /1000

ps_sat_D = 1e-3 * exp(A_D + (B_D / (C_D + Ts - Tref) ) ) / 1000
ps_sat_E = 1e-3 * exp(A_E + (B_E / (C_E + Ts - Tref) ) ) / 1000
ps_sat_F = 1e-3 * exp(A_F + (B_F / (C_F + Ts - Tref) ) ) / 1000
ps_sat_G = 1e-3 * exp(A_G + (B_G / (C_G + Ts - Tref) ) ) / 1000
ps_sat_H = 1e-3 * exp(A_H + (B_H / (C_H + Ts - Tref) ) ) / 1000

 p_D_s = gamma_D_s * separator_x_D * ps_sat_D
 p_E_s = gamma_E_s * separator_x_E * ps_sat_E
 p_F_s = gamma_F_s * separator_x_F * ps_sat_F
 p_G_s = gamma_G_s * separator_x_G * ps_sat_G
 p_H_s = gamma_H_s * separator_x_H * ps_sat_H

ps_MPa = p_A_s + p_B_s + p_C_s + p_D_s + p_E_s + p_F_s + p_G_s + p_H_s
```

```
stream_8_conc_A = p_A_s / ps_MPa
stream_8_conc_B = p_B_s / ps_MPa
stream_8_conc_C = p_C_s / ps_MPa
stream_8_conc_D = p_D_s / ps_MPa
stream_8_conc_E = p_E_s / ps_MPa
stream_8_conc_F = p_F_s / ps_MPa
stream_8_conc_G = p_G_s / ps_MPa
stream_8_conc_H = p_H_s / ps_MPa

############
# Purge    #
############

# Introduced as independent variable instead of valve position
;define_control(stream_9_flow)

stream_9_conc_A = stream_8_conc_A
stream_9_conc_B = stream_8_conc_B
stream_9_conc_C = stream_8_conc_C
stream_9_conc_D = stream_8_conc_D
stream_9_conc_E = stream_8_conc_E
stream_9_conc_F = stream_8_conc_F
stream_9_conc_G = stream_8_conc_G
stream_9_conc_H = stream_8_conc_H

###############
# Compressor  #
###############

# Instead of compressor recycle valve

  stream_8_T = Ts * (pm_MPa / ps_MPa)^( (1 - s_kap) / s_kap )


######################
#From Stripper zone #
######################

Vstr_liq =  (stripper_NG + stripper_NH) / rho_liq_stripper /(stream_11_conc_G +
stream_11_conc_H)

phi_D = PC_1 * (Tstr - Tref)^3 + PC_6  * (Tstr - Tref)^2 + PC_11 * (Tstr -
Tref) + PC_16
phi_E = PC_2 * (Tstr - Tref)^3 + PC_7  * (Tstr - Tref)^2 + PC_12 * (Tstr -
Tref) + PC_17
```

```
phi_F = PC_3 * (Tstr - Tref)^3 + PC_8  * (Tstr - Tref)^2 + PC_13 * (Tstr -
Tref) + PC_18
phi_G = PC_4 * (Tstr - Tref)^3 + PC_9  * (Tstr - Tref)^2 + PC_14 * (Tstr -
Tref) + PC_19
phi_H = PC_5 * (Tstr - Tref)^3 + PC_10 * (Tstr - Tref)^2 + PC_15 * (Tstr -
Tref) + PC_20

dNstr_G = (1 - phi_G) * (separator_x_G * stream_10_flow + stream_4_conc_G *
stream_4_flow) - stream_11_conc_G * stream_11_flow
dNstr_H = (1 - phi_H) * (separator_x_H * stream_10_flow + stream_4_conc_H
*stream_4_flow) - stream_11_conc_H * stream_11_flow

stream_5_flow = stream_10_flow + stream_4_flow - stream_11_flow - (dNstr_G
 + dNstr_H)

stream_5_conc_A  = phi_A * (stream_4_conc_A * stream_4_flow + separator_x_A *
stream_10_flow) / stream_5_flow
stream_5_conc_B  = phi_B * (stream_4_conc_B * stream_4_flow + separator_x_B *
stream_10_flow) / stream_5_flow
stream_5_conc_C  = phi_C * (stream_4_conc_C * stream_4_flow + separator_x_C *
stream_10_flow) / stream_5_flow
stream_5_conc_D  = phi_D * (stream_4_conc_D * stream_4_flow + separator_x_D *
stream_10_flow) / stream_5_flow
stream_5_conc_E  = phi_E * (stream_4_conc_E * stream_4_flow + separator_x_E *
stream_10_flow) / stream_5_flow
stream_5_conc_F  = phi_F * (stream_4_conc_F * stream_4_flow + separator_x_F *
stream_10_flow) / stream_5_flow
stream_5_conc_G  = phi_G * (stream_4_conc_G * stream_4_flow + separator_x_G *
stream_10_flow) / stream_5_flow
stream_5_conc_H  = phi_H * (stream_4_conc_H * stream_4_flow + separator_x_H *
stream_10_flow) / stream_5_flow

stream_11_conc_A = 0
stream_11_conc_B = 0
stream_11_conc_C = 0

# Conc D - F
stream_11_conc_D = (stream_4_conc_D * stream_4_flow + separator_x_D *
stream_10_flow -
stream_5_conc_D * stream_5_flow ) / stream_11_flow
stream_11_conc_E = (stream_4_conc_E * stream_4_flow + separator_x_E *
stream_10_flow -
stream_5_conc_E * stream_5_flow ) / stream_11_flow
stream_11_conc_F = (stream_4_conc_F * stream_4_flow + separator_x_F *
stream_10_flow -
```

```
stream_5_conc_F * stream_5_flow ) / stream_11_flow


# Conc G, H
stream_11_conc_G  = ((1 - stream_11_conc_D - stream_11_conc_E -
stream_11_conc_F) *
stripper_NG) / (stripper_NG + stripper_NH)
stream_11_conc_H  = ((1 - stream_11_conc_D - stream_11_conc_E -
stream_11_conc_F) *
stripper_NH) / (stripper_NG + stripper_NH)

##########################
# Mixing Unit           #
##########################

# Molar Balance for components A-H
dmixing_zone_NA = stream_1_flow * stream_1_conc_A + stream_2_flow *
stream_2_conc_A + stream_3_flow * stream_3_conc_A + stream_5_flow *
stream_5_conc_A + stream_8_flow * stream_8_conc_A - stream_6_flow *
stream_6_conc_A

dmixing_zone_NB = stream_1_flow * stream_1_conc_B + stream_2_flow *
stream_2_conc_B + stream_3_flow * stream_3_conc_B + stream_5_flow *
stream_5_conc_B + stream_8_flow * stream_8_conc_B - stream_6_flow *
stream_6_conc_B

dmixing_zone_NC = stream_1_flow * stream_1_conc_C + stream_2_flow *
stream_2_conc_C + stream_3_flow * stream_3_conc_C + stream_5_flow *
stream_5_conc_C + stream_8_flow * stream_8_conc_C - stream_6_flow *
stream_6_conc_C

dmixing_zone_ND = stream_1_flow * stream_1_conc_D + stream_2_flow *
stream_2_conc_D + stream_3_flow * stream_3_conc_D + stream_5_flow *
stream_5_conc_D + stream_8_flow * stream_8_conc_D - stream_6_flow *
stream_6_conc_D

dmixing_zone_NE = stream_1_flow * stream_1_conc_E + stream_2_flow *
stream_2_conc_E + stream_3_flow * stream_3_conc_E + stream_5_flow *
stream_5_conc_E + stream_8_flow * stream_8_conc_E - stream_6_flow *
stream_6_conc_E

dmixing_zone_NF = stream_1_flow * stream_1_conc_F + stream_2_flow *
stream_2_conc_F + stream_3_flow * stream_3_conc_F + stream_5_flow *
stream_5_conc_F + stream_8_flow * stream_8_conc_F - stream_6_flow *
stream_6_conc_F
```

```
dmixing_zone_NG = stream_1_flow * stream_1_conc_G + stream_2_flow *
stream_2_conc_G + stream_3_flow * stream_3_conc_G + stream_5_flow *
stream_5_conc_G + stream_8_flow * stream_8_conc_G - stream_6_flow *
stream_6_conc_G

dmixing_zone_NH = stream_1_flow * stream_1_conc_H + stream_2_flow *
stream_2_conc_H + stream_3_flow * stream_3_conc_H + stream_5_flow *
stream_5_conc_G + stream_8_flow * stream_8_conc_H - stream_6_flow *
stream_6_conc_H


# Energy balance for the mixing zone
  mixing_zone_Ncp =    mixing_zone_NA * cp_vap_A + mixing_zone_NB *
   cp_vap_B +
mixing_zone_NC * cp_vap_C + mixing_zone_ND * cp_vap_D + mixing_zone_NE *
cp_vap_E + mixing_zone_NF * cp_vap_F + mixing_zone_NG * cp_vap_G +
 mixing_zone_NH * cp_vap_H

#Streem Cp
  stream_5_cp = stream_5_conc_A * cp_vap_A + stream_5_conc_B * cp_vap_B +
stream_5_conc_C * cp_vap_C + stream_5_conc_D * cp_vap_D + stream_5_conc_E
 * cp_vap_E + stream_5_conc_F * cp_vap_F + stream_5_conc_G * cp_vap_G +
  stream_5_conc_H * cp_vap_H
stream_8_cp = stream_8_conc_A * cp_vap_A + stream_8_conc_B * cp_vap_B +
stream_8_conc_C * cp_vap_C + stream_8_conc_D * cp_vap_D + stream_8_conc_E *
cp_vap_E + stream_8_conc_F * cp_vap_F + stream_8_conc_G * cp_vap_G +
stream_8_conc_H * cp_vap_H

# parts of right hand side of equation
  mix_enth_1 = stream_1_flow * stream_1_cp * (stream_1_T - Tm)
  mix_enth_2 = stream_2_flow * stream_2_cp * (stream_2_T - Tm)
  mix_enth_3 = stream_3_flow * stream_3_cp * (stream_3_T - Tm)
  mix_enth_5 = stream_5_flow * stream_5_cp * (Tstr       - Tm)
  mix_enth_8 = stream_8_flow * stream_8_cp * (stream_8_T - Tm)

# Right hand side of equation
 mix_rhs = (mix_enth_1 + mix_enth_2 + mix_enth_3 + mix_enth_5 + mix_enth_8) /
mixing_zone_Ncp

# Temperature in the mixing zone
   dTm = mix_rhs

#################
#   Reactor    #
```

```
################

reactor_R1 = alpha_1 * Vr_vap * exp(44.06 - (42600/(Rkcal * Tr))  ) *
 ((p_A_r*1000)^1.080) *
((p_C_r*1000)^0.311) * ((p_D_r*1000)^0.874) / 3600

reactor_R2 = alpha_2 * Vr_vap * exp(10.27 - (19500/(Rkcal * Tr))  ) *
 ((p_A_r*1000)^1.150) *
((p_C_r*1000)^0.370) * ((p_E_r*1000)^1.000) / 3600

reactor_R3 = alpha_3 * Vr_vap * exp(59.50 - (59500/(Rkcal * Tr))  ) *
 (p_A_r*1000) * ( 0.77 *
(p_D_r*1000) + (p_E_r*1000)) / 3600

reactor_conv_rate_A = mue_A_1 * reactor_R1 + mue_A_2 * reactor_R2 +
mue_A_3 *  reactor_R3
reactor_conv_rate_B = mue_B_1 * reactor_R1 + mue_B_2 * reactor_R2 +
 mue_B_3 * reactor_R3
reactor_conv_rate_C = mue_C_1 * reactor_R1 + mue_C_2 * reactor_R2 +
 mue_C_3 * reactor_R3
reactor_conv_rate_D = mue_D_1 * reactor_R1 + mue_D_2 * reactor_R2 +
 mue_D_3 * reactor_R3
reactor_conv_rate_E = mue_E_1 * reactor_R1 + mue_E_2 * reactor_R2 +
 mue_E_3 * reactor_R3
reactor_conv_rate_F = mue_F_1 * reactor_R1 + mue_F_2 * reactor_R2 +
 mue_F_3 * reactor_R3
reactor_conv_rate_G = mue_G_1 * reactor_R1 + mue_G_2 * reactor_R2 +
 mue_G_3 * reactor_R3
reactor_conv_rate_H = mue_H_1 * reactor_R1 + mue_H_2 * reactor_R2 +
 mue_H_3 * reactor_R3


dreactor_NA = stream_6_flow * stream_6_conc_A - stream_7_flow *
 stream_7_conc_A + reactor_conv_rate_A
dreactor_NB = stream_6_flow * stream_6_conc_B - stream_7_flow *
 stream_7_conc_B + reactor_conv_rate_B
dreactor_NC = stream_6_flow * stream_6_conc_C - stream_7_flow *
 stream_7_conc_C + reactor_conv_rate_C
dreactor_ND = stream_6_flow * stream_6_conc_D - stream_7_flow *
 stream_7_conc_D + reactor_conv_rate_D
dreactor_NE = stream_6_flow * stream_6_conc_E - stream_7_flow *
 stream_7_conc_E + reactor_conv_rate_E
dreactor_NF = stream_6_flow * stream_6_conc_F - stream_7_flow *
 stream_7_conc_F + reactor_conv_rate_F
dreactor_NG = stream_6_flow * stream_6_conc_G - stream_7_flow *
```

```
 stream_7_conc_G + reactor_conv_rate_G
dreactor_NH = stream_6_flow * stream_6_conc_H - stream_7_flow *
 stream_7_conc_H + reactor_conv_rate_H

reactor_Ncp =   reactor_NA * cp_vap_A + reactor_NB * cp_vap_B + reactor_NC
 * cp_vap_C + reactor_ND * cp_liq_D + reactor_NE * cp_liq_E + reactor_NF *
  cp_liq_F + reactor_NG * cp_liq_G + reactor_NH * cp_liq_H
stream_6_cp = stream_6_conc_A * cp_vap_A + stream_6_conc_B * cp_vap_B +
stream_6_conc_C * cp_vap_C + stream_6_conc_D * cp_vap_D + stream_6_conc_E
*cp_vap_E + stream_6_conc_F * cp_vap_F + stream_6_conc_G * cp_vap_G +
stream_6_conc_H * cp_vap_H

  H_A = cp_vap_A * (Tr - Tref)
  H_B = cp_vap_B * (Tr - Tref)
  H_C = cp_vap_C * (Tr - Tref)
  H_D = cp_vap_D * (Tr - Tref)
  H_E = cp_vap_E * (Tr - Tref)
  H_F = cp_vap_F * (Tr - Tref)
  H_G = cp_vap_G * (Tr - Tref)
  H_H = cp_vap_H * (Tr - Tref)

delt_Hr_1 = (mue_A_1 * H_A + mue_C_1 * H_C + mue_D_1 * H_D + mue_G_1 * H_G
 - 136033.04e0) / 1000
delt_Hr_2 = (mue_A_2 * H_A + mue_C_2 * H_C + mue_E_2 * H_E + mue_H_2 * H_H
 - 93337.9616e0) / 1000
delt_Hr_3 = (mue_A_3 * H_A + mue_D_3 * H_D + mue_E_3 * H_E + mue_F_3 * H_F

reactor_exoth_heat = reactor_R1 * delt_Hr_1 + reactor_R2 * delt_Hr_2 +
 reactor_R3 * delt_Hr_3

# --- Heat Exchanger ------------------
# Heat exchange with cooling water
  T_CWSr_in = 0.308e+03

# Assumption - from measurement data
   T_CWSr_out = 0.367599e+03

# Preparation for delt_T_log
  delt_T1_reactor = Tr - T_CWSr_in
  delt_T2_reactor = Tr - T_CWSr_out

#  delt_T1_reactor/(delt_T2_reactor+0.000000001))^delt_T_log
 = exp(delt_T1_reactor - delt_T2_reactor)

   delt_T_log = (delt_T1_reactor -
```

```
   delt_T2_reactor)/(ln(delt_T1_reactor/(delt_T2_reactor)))

 Qr = reactor_m_CWS * cp_cw * (T_CWSr_out - T_CWSr_in) / 1000

# Delt T log try
   Qr = UA * delt_T_log / 1000

reactor_rhs = ((1/1000) * stream_6_flow * stream_6_cp * (Tm - Tr) - Qr -
 reactor_exoth_heat) / reactor_Ncp

# Temperature in the reaction zone
  dTr = reactor_rhs

#################
# Separator    #
#################

dseparator_NA = stream_7_flow * stream_7_conc_A - (stream_8_flow +
 stream_9_flow) * stream_8_conc_A - stream_10_flow * separator_x_A
dseparator_NB = stream_7_flow * stream_7_conc_B - (stream_8_flow +
 stream_9_flow) * stream_8_conc_B - stream_10_flow * separator_x_B
dseparator_NC = stream_7_flow * stream_7_conc_C - (stream_8_flow +
 stream_9_flow) * stream_8_conc_C - stream_10_flow * separator_x_C
dseparator_ND = stream_7_flow * stream_7_conc_D - (stream_8_flow +
 stream_9_flow) * stream_8_conc_D - stream_10_flow * separator_x_D
dseparator_NE = stream_7_flow * stream_7_conc_E - (stream_8_flow +
 stream_9_flow) * stream_8_conc_E - stream_10_flow * separator_x_E
dseparator_NF = stream_7_flow * stream_7_conc_F - (stream_8_flow +
 stream_9_flow) * stream_8_conc_F - stream_10_flow * separator_x_F
dseparator_NG = stream_7_flow * stream_7_conc_G - (stream_8_flow +
 stream_9_flow) * stream_8_conc_G - stream_10_flow * separator_x_G
dseparator_NH = stream_7_flow * stream_7_conc_H - (stream_8_flow +
 stream_9_flow) * stream_8_conc_H - stream_10_flow * separator_x_H

separator_Ncp = separator_NA * cp_vap_A + separator_NB * cp_vap_B +
 separator_NC * cp_vap_C + separator_ND * cp_liq_D + separator_NE *
  cp_liq_E + separator_NF * cp_liq_F + separator_NG * cp_liq_G +
  separator_NH * cp_liq_H

stream_7_cp   = stream_7_conc_A * cp_vap_A + stream_7_conc_B * cp_vap_B +
stream_7_conc_C * cp_vap_C + stream_7_conc_D * cp_vap_D + stream_7_conc_E
 *cp_vap_E + stream_7_conc_F * cp_vap_F + stream_7_conc_G * cp_vap_G +
  stream_7_conc_H * cp_vap_H

 HoVs = (separator_x_D * stream_10_flow * M_D * H_vap_D + separator_x_E *
```

```
 stream_10_flow * M_E * H_vap_E + separator_x_F * stream_10_flow * M_F *
  H_vap_F + separator_x_G * stream_10_flow * M_G * H_vap_G + separator_x_H
   * stream_10_flow * M_H * H_vap_H) / 1000

# Heat exchange with cooling water
T_CWSs_in = 0.313e+03

# Assumption - from measurement data
   T_CWSs_out = 350.447

 Qs = separator_m_CWS * cp_cw * (T_CWSs_out - T_CWSs_in) / 1000

# Preparation for delt_T_log
  delt_T1_separator = Ts - T_CWSs_in
  delt_T2_separator = Ts - T_CWSs_out

delt_T_log_sep = (delt_T1_separator -
delt_T2_separator)/(ln(delt_T1_separator/(delt_T2_separator)))

Qs = UA_sep * (delt_T_log_sep) / 1000

separator_rhs = ( (1/1000) * stream_7_flow * stream_7_cp * (Tr - Ts) +
 HoVs - Qs)/ separator_Ncp

# Temperature in the separation zone
   dTs = separator_rhs

################
# Stripper    #
################

dstripper_NG = dNstr_G
dstripper_NH = dNstr_H

stripper_Ncp = stripper_NG * cp_liq_G + stripper_NH * cp_liq_H
stream_10_cp = separator_x_A * cp_liq_A + separator_x_B * cp_liq_B +
 separator_x_C * cp_liq_C + separator_x_D * cp_liq_D + separator_x_E *
 cp_liq_E + separator_x_F * cp_liq_F + separator_x_G * cp_liq_G +
  separator_x_H * cp_liq_H

HoVstr =  ((stream_5_conc_D * stream_5_flow - stream_4_conc_D *
 stream_4_flow) * M_D * H_vap_D + (stream_5_conc_E * stream_5_flow -
 stream_4_conc_E * stream_4_flow) * M_E * H_vap_E + (stream_5_conc_F *
 stream_5_flow - stream_4_conc_F * stream_4_flow) * M_F *
```

```
H_vap_F + (stream_5_conc_G * stream_5_flow - stream_4_conc_G *
  stream_4_flow) * M_G * H_vap_G + (stream_5_conc_H * stream_5_flow -
  stream_4_conc_H * stream_4_flow) * M_H * H_vap_H) / 1000

 Qstr = 2.258717 * stripper_m_CWS

stripper_rhs =( (1/1000) * stream_10_flow * stream_10_cp * (Ts - Tstr) +
 (1/1000) * stream_4_flow  * stream_4_cp  * (stream_4_T - Tstr) - HoVstr +
  Qstr)/ stripper_Ncp


# Temperature in the stripper zone
  dTstr = stripper_rhs

XMEAS_6  = stream_6_flow / 44.79*3600
XMEAS_7  = pr_MPa*1000  - 101
XMEAS_8  = 5.263*Vr_liq - 12.105
XMEAS_9  = Tr - Tref
```

Nomenclature

$-\Delta H$	propagation reaction heat of reaction [case study 3]
$[S]$	Concentration of the S specie in the reactor, where S = *solids*, *others*, $Pr(propionate)$, $But(butyrate)$, $Ac(acetate)$, CH_4 (methane) [case study 1]
$[S]_0$	Concentrations of the S specie in the reactor at the initial time (i.e., t = 0) [case study 1]
$[S]_f$	Feed concentrations of the S specie in the reactor [case study 1]
$\alpha_{i,j}$	Recovery factor of component i w.r.t. j stream [case study 4]
$\bar{m}$	mean segment number in the system [PC-SAFT EOS]
β	Mean path of vapour molecule [case study 2]
$\dot{Q}$	Heat duty, kW [case study 4]
ϵ	Depth of pair potential [PC-SAFT EOS]
ϵ_k	Reaction constant $\mathrm{kmol\,h^{-1}\,m^{-3}}$ [case study 4]
η	Packing fraction [PC-SAFT EOS]
η_i	Liquid viscosity of the i-th component [case study 2]
γ_i	Activity coefficient of the i-th component [case study 2]
κ	Anisotropy degree of the vapour [case study 2]
λ	Purge factor [case study 4]
λ	Reduced well width of square-well potential [PC-SAFT EOS]
λ	eigenvalue [case study 3]
λ_i	Thermal conductivity of the i-th component [case study 2]
Φ_i	Split factor for component i in the stripper [case study 4]
π	Pi [case study 2]
ρ	Total number density of molecules [PC-SAFT EOS]
ρ	reaction mixture density [case study 3]
ρ_i	Density of the i-th component [case study 2]

ρ_w cooling fluid density [case study 3]

σ Purge relationship $1/(1-\lambda)$ [case study 4]

σ Segment diameter [PC-SAFT EOS]

$\upsilon_{i,k}$ Stoichiometric coefficient of component i in reaction k [case study 4]

ζ_n Abbreviation $(n = 0, ..., 3)$ [PC-SAFT EOS]

A Heat-transfer area [case study 3]

A Helmholtz free energy [PC-SAFT EOS]

A_1 Helmholtz free energy of first-order perturbation term [PC-SAFT EOS]

A_2 Helmholtz free energy of second-order perturbation term [PC-SAFT EOS]

A_I Initiation reaction Arrhenius pre-exponential factor [case study 3]

A_p propagation reaction Arrhenius pre-exponential factor [case study 3]

a_{01}, a_{02}, a_{03} model constants [PC-SAFT EOS]

A_{fm} Monomer transfer reaction Arrhenius pre-exponential factor [case study 3]

$a_{j(m)}$ Functions [PC-SAFT EOS]

A_{tc} coupling termination reaction Arrhenius pre-exponential factor [case study 3]

A_{td} disproportionation termination reaction Arrhenius pre-exponential factor [case study 3]

b_{01}, b_{02}, b_{03} model constants [PC-SAFT EOS]

$C(z)$ Total composition [case study 2]

C_I initiator concentration [case study 3]

$C_i(y, z)$ Composition profile of each compound [case study 2]

C_m monomer concentration [case study 3]

C_p reaction mixture heat capacity [case study 3]

$C_{i,0}$ Feed composition of each compound [case study 2]

C_{Iin} initiator feed stream concentration [case study 3]

C_{min} monomer feed stream concentration [case study 3]

$C_{p,i}$	Thermal capacity of the i-th component [case study 2]
C_{pw}	cooling fluid heat capacity [case study 3]
$calc$	subscript, calculated property) [PC-SAFT EOS]
$crit$	subscript, critical property) [PC-SAFT EOS]
d	Distance between evaporator - condenser [case study 2]
d	Temperature-dependent segment diameter [PC-SAFT EOS]
D_0	molar concentration of polymer death chains [case study 3]
D_1	mass concentration of polymer death chains [case study 3]
D_i	Diffusion coefficient of the i-th component [case study 2]
$disp$	subscript, contribution due to dispersive attraction) [PC-SAFT EOS]
E_I	initiation reaction activation energy [case study 3]
E_p	propagation reaction activation energy [case study 3]
E_{fm}	monomer transfer reaction activation energy [case study 3]
E_{tc}	coupling termination reaction activation energy [case study 3]
E_{td}	disproportionation termination reaction activation energy [case study 3]
exp	subscript, experimental property) [PC-SAFT EOS]
F	Inlet and outlet flow rate. [case study 1]
F	Surface ratio [case study 2]
F	monomer feed stream volumetric flow rate [case study 3]
f	sum of square residuals
F_I	initiator feed stream volumetric flow rate [case study 3]
F_j	Flowrate of stream j, $\mathrm{kmol\,s^{-1}}$ [case study 4]
F_{cw}	cooling fluid feed stream volumetric flow rate [case study 3]
g	Coefficient (mmol H_2 . mg^{-1} COD H_2) [case study 1]
g	Gravitational constant [case study 2]
g^{hc}	Average radial distribution function of hard-chain fluid [PC-SAFT EOS]
$g^{hc}_{\alpha\beta}$	Site-site radial distribution function of hard-chain fluid [PC-SAFT EOS]

$h_1(z)$	Thickness film profile [case study 2]
H_i	Enthalpy of component i, kJ [case study 4]
$H_{i,vap}$	Heat of evaporation of the i-th component [case study 2]
hc	subscript, residual contribution of hard-chain system) [PC-SAFT EOS]
hs	subscript, residual contribution of hard-sphere system) [PC-SAFT EOS]
i	Component index [case study 4]
$I(z)$	Total exit flow compound [case study 2]
$I_i(z)$	Exit flows of each compound [case study 2]
$I_{i,0}$	Feed Flow of each compound [case study 2]
id	subscript, ideal gas contribution) [PC-SAFT EOS]
Im_{ij}	Imaginary part of the eigenvalue j for the steady state i [case study 3]
j	Stream index [case study 4]
K	K factor [PC-SAFT EOS]
k	Boltzmann constant [PC-SAFT EOS]
k	Reaction index [case study 4]
$k_i(z$	Effective evaporation rate of each compound [case study 2]
$k_{s,i}$	alf saturation constant (i = 2,. . . , 6) [case study 1]
kij	Binary interaction parameter [PC-SAFT EOS]
L	Evaporator length [case study 2]
M	molar mass [PC-SAFT EOS]
m	Mixing zone [case study 4]
m	Number of segments per chain [PC-SAFT EOS]
M_m	monomer molecular weight [case study 3]
Mw, i	Molar weight of the i-th component [case study 2]
N	Number of compounds [case study 2]
N	total number of molecules [PC-SAFT EOS]
n	Number of intermolecular collisions [case study 2]

N_{DOF} Degree of freedom

$N_{i,j}$ Hold-up of component i, kmol_i [case study 4]

NB Percentage of the non-biodegradable others concentration. [case study 1]

P Pressure [PC-SAFT EOS]

P System Pressure [case study 2]

P/P_{ref} Pressure system effect [case study 2]

p_i Partial pressure of component i [case study 4]

P_{H_2} Pressure of H_2 [case study 1]

$P_{i,vap}$ Vapour pressure of the i-th component [case study 2]

$P_{T,r}$ Total pressure in the reactor, kPa [case study 4]

PM polymer mean molecular weight [case study 3]

R Evaporator radius [case study 2]

R Gas constant [PC-SAFT EOS]

R universal ideal gas law constant [case study 3]

r Radial distance between two segments [PC-SAFT EOS]

r Reactor [case study 4]

R_g Gas constant (0.082057 $l.atm.mol^{-1}.K^{-1}$) [case study 1]

r_i Reaction rate (i = 1,. . . , 6) [case study 1]

Re_{ij} Real part of the eigenvalue j for the steady state i [case study 3]

Rg Universal gas constant [case study 2]

s Separator [case study 4]

s_1 Constant defining the pair potential [PC-SAFT EOS]

sat subscript, property at saturation condition) [PC-SAFT EOS]

str Stripper [case study 4]

T Reactor temperature [case study 1]

T Temperature [PC-SAFT EOS]

T reactor temperature [case study 3]

t time [case study 3]

$T(y,z)$ Temperature profile [case study 2]

T_F Feed Temperature [case study 2]

T_j cooling fluid temperature [case study 3]

T_{in} monomer and initiator feed stream temperature [case study 3]

T_{w0} cooling fluid feed stream temperature [case study 3]

T_{w1} Evaporator wall temperature [case study 2]

T_{w2} Condenser wall temperature [case study 2]

U global heat-transfer coefficient [case study 3]

$u(r)$ Pair potential function [PC-SAFT EOS]

$u_{max,i}$ Maximum specific substrate utilization rate (i = 2,. . . , 6) [case study 1]

UA Specific heat transfer rate coefficient, kW K^{-1} [case study 4]

V Molar volume [PC-SAFT EOS]

V covariance Matrix

V reactor volume [case study 3]

$v(y,z)$ Velocity profile [case study 2]

V_0 cooling fluid volume [case study 3]

V_G Volume of gas phase in batch reactor l [case study 1]

V_L Volume of liquid phase in batch reactor l [case study 1]

V_R Reactor volume [case study 1]

$v_{i,j}$ Stoichiometric coefficients (i = 1,. . . , 6 and j = 1,. . . , 7) [case study 1]

X monomer conversion [case study 3]

x Reduced radial distance between two segments [PC-SAFT EOS]

x_i Mole fraction of component i [PC-SAFT EOS]

X_T Biomass concentration [case study 1]

$x_{i,j}$ Mole fraction in the liquid of component i in stream j [case study 4]

y radial coordinate [case study 2]

$y_{i,j}$	Mole fraction in the vapour of component i in stream j [case study 4]
Z	Compressibility factor [PC-SAFT EOS]
z	axial coordinate [case study 2]

References

Allan, B. and Westerberg, A.: 1999, Anonymous class in declarative process modeling, *Ind. Eng. Chem. Res.* **38**, 692–704.

Anderson, D., Tannehill, J. and Pletcher, R.: 1992, *Computational Fluid Mechanics and Heat transfer*, McGraw-Hill, New York, USA.

Andersson, M.: 1994, *Object-Oriented Modelling and Simulation of Hybrid Systems. PhD Thesis.*, PhD thesis, Department of Automatic Control. Lund Institute of Technology. Sweden.

Arczewski, K. and Pietrucha, J.: 1993, *Mathematical Modelling of Mechanical Complex Systems*, Discrete Models, Vol. 1, Ellis Horwood, New York, New York.

Arney, D. and Flaherty, J.: 1990, Adaptive mesh-moving and local refinement method for time-dependent partial differential equations, *ACM Transactions on Mathematical Software* **16**(1), 48–71.

Asbjørnsen, O., Meyassami, B. and Sørlie, C.: 1989, Structuring the knowledge for process modeling from the first principles.., *Paper presented at IAKE'89*, University of Maryland, College Park, USA.

ASPEN, P.: Aspen Technology Inc., Aspen technology inc., *http://www.aspentech.com* . Aspen Technology Inc.

Asprey, S. and Macchietto, S.: 2002, Designing robust optimal experiments, *Journal of Process Control* **12**(4), 545–556.

Baker, H., Chen, M., Grant, P., Jobling, C. and Townsend, P.: 1993, Open architecture for computer-aided control engineering, *IEEE Control System* **Apr.**, 17–27.

Balci, O.: 1986, Requirements for model development environments., *Computers and Opt. Res.* **13**(1), 53–67.

Bär, M. and Zeitz, M.: 1990, A knowledge-based flowsheet oriented user interface for a dynamic process simulator., *Computers and Chemical Engineering* **14**, 1275–1283.

Bard, Y.: 1974, *Nonlinear Parameter Stimation.*, Academic Press, New York, USA.

Batistella, C. and Maciel, M.: 1996, Modelling, simulation and analysis of molecular distillators: Centrifugal and falling film., *Comp. Chem. Eng. Suppl.* pp. S19–S24.

Bezzo, F., Macchietto, S. and Pantelides, C.: 2000, A general framework for the integration of computational fluid dynamics and process simulation., *Computers and Chemical Engineering* **2-7**, 653–658.

Biegler, L.: 1989, Chemical processes simulation, *Chemical Engineering Progress* **Oct.**, 50–61.

Bieszczad, J., Koulouris, A. and Stephanopoulos, G.: 1999, Model.la: A phenomena-based modeling environment for computer-aided process design., *Proc. Foundations of Computer-Aided Process Design*, FOCAPD'99, Breckenridge, CO.

Bogusch, R., Lohmann, B. and Marquardt, W.: 2001, Computer-aided process modeling with modkit, *Computers and Chemical Engineering* **25**(1), 963–995.

Boston, J., Brit, H. and Tayyabkhan, M.: 1993, Software: Tackling tougher task, *Chemical Engineering Progress* **Nov.**, 38–49.

Brenan, K., Campbell, S. and Petzold, L.: 1989, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, Elsevier, Amsterdam.

Byrne, G., Hindmarsh, A., Jackson, K. and Brown, H.: 1977, A comparison of two ode codes gear and episode, *Computers and Chemical Engineering* **1**(2), 125–131.

Cameron, I. and Gani, R.: 1988, Adaptive runge-kutta algorithms for dynamic simulation, *Computers and Chemical Engineering* **12**(7), 705–717.

CAPE-OPEN, C.: http://www.colan.org, Cape-open laboratories, *http://www.colan.org* . network available.

Carey, G. and Finlayson, B.: 1975, Orthogonal collocation on finite elements, *Chemical Engineering Science* **30**(5), 587–596.

Carver, M.: 1981, Method of lines solution of differential equations - fundamental principles and recent extensions, *in* R. Mah and W. D. Seider (eds), *Foundations of Computer-Aided Process Design*, New-york, New Hampshire, pp. 369–402.

Cavani, F. and Trinfir, F.: 1995, Alternative process for the production of styrene, *Applied Catalysis* **133**(2), 219–239.

CHAM, l.: 1987, *The PHOENICS beginners guide Release 1.4*, Wimbledon, U.K.

Cordiner, J.: 2004, Challenges for the pse community in formulations, *Computers and Chemical Engineering* **29**(1), 83–92.

Cvengroš, J., Lutišan, J., and Micov, M.: 2000, Feed temperature influence on the efficiency of a molecular evaporator., *Comp. Chem. Eng. Suppl.* **78**, 61–67.

Cvengroš, J., Micov, M., and Lutišan, J.: 2000, Modelling of fractionation in a molecular evaporator with divided condenser., *Chem. Eng. Proc.* **39**, 191–199.

DIAS: http://www.dis.anl.gov/DIAS, Modilling environment, *http://www.dis.anl.gov/DIAS/index.html* . Internert page.

Downs, J. and Vogel, E.: 1990, A plant-wide industrial process control problem, *AIChE Annual Meeting*, Chicago, IL, p. Paper 24a.

Downs, J. and Vogel, E.: 1993, A plant-wide industrial process control problem, *Computers and Chemical Engineering* **17**(3), 245–255.

FAMAS: http://www.famas.tudelft.nl, Fams modelling framework, *http://www.famas.tudelft.nl* . Internert page.

Fikes, R. and Kehler, T.: 1985, The rol of frame-based representation in reasoning., *ACM Commun.* **28**, 904–920.

Finlayson, B.: 1980, Orthogonal collocation on finite elements-progress and potential, *Mathematics and Computers in Simulation* **22**(1), 11–17.

Foss, B., Lohmann, B. and Marquardt, W.: 1998, A field study of the industrial modeling process, *Journal of Process Control* **5/6**(1), 325–338.

Froment, G. and Bischoff, K.: 1990, *Chemical Reactor Analysis and Design*, second print edn, John Wiley and Sons Ltd (March 21, 1990), New York.

Gani, R.: 2002, Icas documentation. CAPEC, Technical University of Denmark.

Gani, R.: 2004, Computer-aided methods and tools for chemical product design, *Chemical Engineering Research & Design* **82**(A11), 1494–1504.

Gani, R., Hytoft, G., Jaksland, C. and Jensen, A.: 1997, An integrated computer aided system for integrated design of chemical processes, *Computers and Chemical Engineering* **21**(10), 1135–1146.

Geoffrion, A.: 1989, Computer-based modeling environments., *European Journal of Operation Research* **41**, 33–43.

Gershenfeld, N.: 1999, *The Nature of Mathematical Modeling*, Modelling, 1st. edn, Cambrige University Press, Cambrige, U.S.A.

Gilles, E., Gawthrop, P. and Weiss, M.: 1998, Network theory for chemical processes., *Chemical Engineering Technology* **21**, 121132.

Gross, J. and Sadowski, G.: 2001, Pc-saft, *Perturbed-Chain SAFT: An Equation of state Based on a Perturbation Theory fro Chain Molecules.* **40**, 1244–1260.

Guide, A. P. U.: Aspen Tech., *ASPEN PLUS User Guide*, Aspen Tech., Cambridge, MA.

Hangos, K. and Cameron, I.: 2001, *Process Modelling and Model Analysis*, Process Systems Engineering, 1st. edn, Academic Press, Harcourt Place, 32 Jamestown Road, London NW1 7BY, UK.

Heydweiller, J., Sincovec, R. and Fan, L.: 1977, Dynamic simulation of chemical process described by distributed and lumped parameter models, *Computers and Chemical Engineering* **1**(1), 125–131.

Horton, I.: 1998, *Beginning Visual C++ 6*, Wrox Press Ltd, Arden House. 1102 Warwick Road, Acock's Green, Birmingham B27 6BH, UK.

Jarvis, R. and Pantelides, C.: 1991, Robust dynamic simulation of chemical engineering processes, *Chemical Engineering Research and Design* **69**(3), 213–214.

Jensen, A. and Gani, R.: 1996, A computer aided system for generation of problem specific process models, *Computers and Chemical Engineering (supplement)* **20**, S145–S150.

Jensen, A. K.: 1998, *Generation of problem Specific Simulation Models within an Integrated Computer Aided System*, PhD thesis, Department of Chemical Engineering, Technical University of Denmark.

Jiménes, E. R.: 2005, *Modelling. Design, Operability and Analysis of Reaction-Separation with Recycle Systems. PhD Thesis.*, PhD thesis, Department of Chemical Engineering, Technical University of Denmark.

Jockenhövel, T., Biegler, L. T. and Wächter, A.: 2003, Dynamic optimization of the Tennessee Eastman process using the OptControlCentre, *Computers and Chemical Engineering* **27**, 1513–1531.

Kawala, Z.: 1976, Evaporation and separation coefficients in high- and medium vacuum., *Chem. Ing. Tech.* **48**, 81–90.

Kawala, Z., and Stephan, K.: 1989, Evaporation rate and separation factor of molecular distillation in a falling film apparatus., *Chem. Eng. Technol.* **12**, 406–413.

Kristensen, M.: 2004, *Parameter estimation in nonlinear dynamic systems. master's thesis.*, Master's thesis, Department of Chemical Engineering, Technical University of Denmark, Denmark.

Kröner, A., Holl, P., Marquardt, W. and Gilles, E.: 1990, Diva—an open architecture for dynamic simulation, *Computers and Chemical Engineering* **14**(11), 1289–1295.

Lakner, R., Hangos, K. and Cameron, I.: 2005, On minimal models of process systems, *Chemical Engineering Science* **60**, 1127–1142.

Le-Lann, J. and Sargousse, M.: 1996, Reseda: Solution of algebraic differential systems, user manual, *Technical Report LGC-UMR 5503, INP*, Stanford University.

Lohmann, B. and Marquardt, W.: 1996, On the systematization of the process of model development, *Computers and Chemical Engineering* **3**, S213–S218.

Lund, P.: 1992, *An Object-Oriented Environment for Process Modeling and simulation. PhD Thesis.*, PhD thesis, Laboratory of Chemical Engineering, Norwegian Institute of Technology, Trondheim.

Lutišan, J., Cvengroš, J., and Micov, M.: 2002, Heat and mass transfer in the evaporating film of a molecular evaporator., *Chem. Eng. J.* **85**, 225–234.

Machura, M. and Sweet, R.: 1980, A survey of software for partial differential equations, *ACM trans. Math. Soft.* **6**, 461–488.

Madsen, N. and Sincovec, R.: 1980, Pdecol, general collocation software for partial differential equations, *ACM trans. Math. Soft.* **5**, 326–351.

Manager, M.: Aspen Tech., *Model Manager Reference Manual*, Aspen Tech., Cambridge, MA.

Marquardt, W.: 1991, Dynamic process simulation recent trends and future challenges., *in* A. CACHE, Austin (ed.), *Chemical Process Control CPC-IV*, AIChE Symposium Series 91, Y. Arkun and W. H. Ray, Eds., New York, pp. 131–180.

Marquardt, W.: 1994, Computer-aided generation of chemical engineering process models, *American Institute of Chemical Engineers* **34**(1), 2846.

Marquardt, W.: 1996, Trends in computer-aided process modeling, *Computers and Chemical Engineering* **20**(6/7), 591–609.

Mattson, S. and Anderson, M.: 1993, *OMOLA. An Object-Oriented Modeing Language.*, Process Systems Engineering, M. Jamshidi and C.J. Herget, Eds., Elsevier. Amsterdam.

Mattson, S., Elmquist, H. and Otter, M.: 1998, Physical system modeling with modelica, *Control Engineering Practice,* **6**, 501510.

Micov, M., Lutišan, J., and Cvengroš, J.: 1997, Balance equations for molecular distillation., *Separ. Sci. Technol.* **32**(8), 3051–3066.

Miller, K.: 1981, Moving finite elements. ii, *ACM Transactions on Mathematical Software* **18**(5), 48–71.

Morton, W. and Collingwood, C.: 1998, An equation analyser for process models, *Computers and Chemical Engineering* **22**(4/5), 571–585.

Németh, E., Cameron, I. and Hangos, K.: 2005, Diagnostic goal driven modelling and simulation of multiscale process systems, *Computers and Chemical Engineering* **29**, 783–796.

Nguyen, A., and Goffic, F. L.: 1997, Limits of wiped film short-path distiller., *Chem. Eng. Sci.* **52**(16), 2661–2666.

Nyhoff, L. and Sanford, C.: 1997, *Fortran 90 for Engineers*, second print edn, Prentice-Hall Inc., New Jersey.

Oh, M.: 1995, *Modelling and simulation of combined lumped and distributed processes, PhD Thesis,*, PhD thesis, Imperial College, Londres U.K.

Oh, M. and Pantelides, C.: 1995, A modelling and simulation language for combined lumped and distribited parameter systems, *Computers and Chemical Engineering* **20**(6/7), 611–633.

Olah, G.: 1964, *Friedel-Crafts and Related Reactions*, Wiley-Interscience, New York.

Pantakar, V.: 1980, *Numerical heat transfer and Fluid flow*, McGraw-Hill, New York, USA.

Pantelides, C. and Barton, P.: 1993, Equation-oriented dynamic simulation —current status and future perspectives., *Computers and Chemical Engineering* **17S**, S263–S285.

Pantelides, C. and Barton, P.: 1994, Multipurpose process modeling environments, *Foundations Computer Aided Design Conference.*, Computer-Aided Process Design, FOCAPD'94, Snowmasss CO.

Pantelides, C. and Brit, H.: 1995, Multipurpose process modeling environments, *Foundations Computer Aided Design Conference.*, Computer-Aided Process Design, FOCAPD'95, Snowmasss CO, pp. 128–141.

Perkins, J. and Barton, G.: 1987, Modelling and simulation in process operation, *Foundations of Computer Aided Process Operations. Proceedings of the First International Conference*, Computer Aided Process Operations, FOCAPD'87, pp. 287–316.

Perkins, J., Sargent, R., Vázquez-Román, R. and Cho, J.: 1996, Computer generation of process models., *Computers and Chemical Engineering* **20**, 635–639.

Petzold, L.: 1982, Description of dassl: A differential/algebraic system solver, *10th IMACS World Congress on System Simulation and Scientific Computation.* **1**(2), 430–432.

Pfeiffer, T. and Marquardt, W.: 1996, Symbolic semi-discretization of partial differential equation systems, *Mathematics and computers in simulation* **42**, 617–628.

Piela, P., Epperly, T., Westerberg, K. and Westerberg, A.: 1991, Ascend —an object-oriented computer aided environment for modeling and analysis: The modeling language, *Computers and Chemical Engineering* **15**, 53–72.

Piela, P., McKelvey, R., and Westerberg, A.: 1992, An introduction to ascend: its language and interactive environment., *Journal Man. Info Sci.* **9**, 91–121.

Ponton, J., Gawthrop, P. and Weiss, M.: 1999, Systematic construction of dynamic models for phase equilibrium processes, *Computers and Chemical Engineering* **15**, 803808.

Preisig, H.: 1995, Modeller. an object-oriented computer-aided modelling tool., *in* F. of Computer-Aided Process Design (ed.), *L.T Biegler and M.F: Doherty (Eds.)*, AIChE Symposium Series 95, AIChE Symposium Series, pp. 328–331.

Preisig, R. and Marquardt, W.: 2001, Revisar titulo, *Computers and Chemical Engineering* **25**, 963995.

Pyton: http://www.python.org, Python documentation, *http://www.python.org* . Internert page.

Ramkrishna, D.: 1985, The status of population balances, *Rev. in Chem. Engng* **3**, 49–90.

Ray, W. and Villa, C.: 2000, Nonlinear dynamics found in polymerization processes - a review, *Chemical Engineering Science* **55**, 275 290.

Reddy, K. and Doraiswamy, L.: 1967, Estimating liquid diffusivity., *Ind. Eng. Chem. Fundam.* **6**, 77–84.

Ricker, N. and Lee, J.: 1995, Nonlinear modeling and state estimation for the Tennessee Eastman Challenge Process, *Computers and Chemical Engineering* **19**(9), 983–1005.

Rico-Ramirez, V.: 1998, *Representation, Analysis and Solution of Conditional Models in an Equation-Based Environment. PhD Thesis.*, PhD thesis, Carnegie Mellon University, Pittsburgh, Pennsylvania.

Russel, B., Henriksen, J., Jørgensen, S. and Gani, R.: 2002, Integration of design and control through model analysis, *Computers and Chemical Engineering* **26**(1), 213225.

Sales-Cruz, M. and Gani, R.: 2003, *Computer-Aided Chemical Engineering: Dynamic Model Development*, first edition edn, Eds. S.P. Asprey and S. Macchietto, Elsevier, Amsterdam.

Sanz-Serna, J. and Verwer, J.: 1989, Stability and convergence at the pde/stiff ode interface, *Applied Numerical Mathematics* **5**(1-2), 117–132.

Schiesser, W.: 1991, *The Numerical Method of lines*, Academic Press, New York, USA.

Schiesser, W.: 1996, Pde boundary conditions from minimum reduction of the pde, *Applied Numerical Mathematics* **20**, 171–179.

Schopfer, G., Yang, A., von Wedel, L. and Marquardt, W.: 2004, Cheops: A tool-integration platform for chemical process modelling and simulation, *Int J Software Tools Transfer* **6**, 186–202.

Seber, G. and Wild, C.: 2003, *Nonlinear Regression*, John Wiley & Sons, New York, USA.

Smith, E.: 1996, *Continuous Process Design Optimization. PhD Thesis.*, PhD thesis, Imperial College of Science and Technology, UK.

Sørlie, C.: 1990, *A Computer Environment for Process Modeling. PhD Thesis.*, PhD thesis, Laboratory of Chemical Engineering, Norwegian Institute of Technology, Trondheim.

Steele, G.: 1990, *Common Lisp. The Language*, 2nd edn. edn, Digital Press.

Stephanopoulos, G., Henning, G. and Leone, H.: 1990, Model.la a modeling language for process engineering-i. the formal framework, *Computers and Chemical Engineering* **14**(8), 813–846.

Stephanopoulos, G., Johnston, J., Kriticos, T., Lakshmanan, R., Mavrovouniotis, M. and Siletti, C.: 1987, Desig-kit: An object-oriented environment for process engineering, *Computers and Chemical Engineering* **11**(6), 655–674.

Stortelder, W.: 1998, *Parameter Estimation in Nonlinear Dynamic System, PhD Thesis*, PhD thesis, National Research Institute for Mathematics and Computer Science. University of Amsterdam.

Strang, G. and Fox, G.: 1973, *An Analysis of the Finite Element Method*, Prentice-Hall, New Jersey, USA.

Tech., A.: Aspen Tech., *SPEEDUP 2000 User Manual*, Aspen Tech., Cambridge, MA.

Toutain, J., Joulia, X., Gourdon, C. and LeLann, J.: 1998, Maxwell-stefan approach coupled drop poulation model for the dynamic simulation of liquid-liquid extraction pulsed column, *in* E. 8 (ed.), *ESCAPE 8*, Brugge, Belgique.

Tränkle, F., Zeitz, M., Ginkel, M. and Gilles, E.: 2000, Promot: A modeling tool for chemical processes., *Mathematical and Computer Modelling of Dynamical Systems* **6**, 283307.

Vázquez-Román, R., King, J. and Bañarez-Alcántara, R.: 1996, Kbmoss: A process engineering modelling support system, *Computers and Chemical Engineering* **20**(Suppl), S309–S314.

Villadsen, J. and Stewart, W.: 1967, Solution of boundary-value problems by orthogonal collocation., *Chemical Engineering Science* **22**(11), 1483–1501.

Westerweele, M., Preisig, H. and Weiss, M.: 1999, Concept and design of modeller, a computer-aided modelling tool., *Computers and Chemical Engineering Supplement* **23**, S751 S754.

Whitney, T., Rode, F. and Tung, C.: 1972, The 'powerful pocketful': an electronic calculator challenges the slide rule., *Reverse Polish Notation* **1**.

Index